Le multimédia et la compression

Stéfane Paris

Le multimédia et la compression

A Tenzing

LAVOISIER
11, rue Lavoisier
75008 Paris

www.hermes-science.com
www.lavoisier.fr

ISBN 978-2-7462-2203-8
ISSN 1242-7691

Printed and bound in England by Antony Rowe Ltd, Chippenham, September 2009.

Le multimédia et la compression

Stéphane Paris

Hermes Science publications

Lavoisier

DIRECTION EDITORIALE JEAN-CHARLES POMEROL

Collection Informatique

Table des matières

Préface

En tant que maître de conférence en informatique, passionné par l'image, j'ai suivi la naissance du multimédia avec intérêt. Et, l'idée d'écrire ce livre s'est imposée assez naturellement : la formation proposée aux étudiants en informatique doit comporter les fondements et les outils du multimédia.

Le statut du multimédia est similaire à celui de l'architecture des ordinateurs. Mise à part quelques exceptions, les informaticiens ne créent pas de nouveaux micoprocesseurs, ne câblent pas de circuits intégrés. Cependant, ils doivent connaître le fonctionnement des machines (présentes et futures) sur lesquelles ils vont programmer.

De même, le multimédia doit faire partie de la culture scientifique informatique. L'informaticien ne peut programmer et gérer le *web* de demain sans connaître les outils qu'il manipule. Ces outils ne sont autres que les audiovisuels, les images, les hypertextes, les graphiques, etc. En un mot : le multimédia.

Cette idée s'est concrétisée quand j'ai proposé un cours de multimédia. Initialement, j'avais envisagé de ne faire que quelques heures. Je voulais proposer aux étudiants un survole du sujet. Mais très vite, ce volume horaire est apparu trop restreint pour répondre à toutes leurs questions. J'ai donc déposé sur mon site Internet des références en adéquation avec le public informatique. Je n'ai, hélas, trouvé qu'un seul livre qui aborde les fondements du multimédia de manière compréhensible – sans prérequis – pour des étudiants informaticiens : *Fundamentals of multimedia*, par Z.N. Li et M.S. Drew, aux éditions Prentice-Hall.

Pour conclure cette préface, je tiens à remercier Guy Bourhis, Francois Brucker, Azzeddine Kaced et Georges Paris pour leurs relectures et leurs critiques ; Mark S. Drew, Ze-Nian Li, Dale Purves, David Salomon et Jacques Weiss pour m'avoir généreusement offert de leurs temps et de leurs matériels ; l'Université Paul Verlaine de

Metz de m'avoir offert un congé d'enseignement qui a facilité la rédaction de ce livre ; les créateurs du logiciel TeXmacs.

Enfin, je remercie tout particulièrement Elisabeth, Evelyne et Sylvie pour leur aide et leur soutien dans ce projet.

Avant-propos

Les deux volumes de ce livre présentent en trois parties l'ensemble des outils actuels (en 2008) du multimédia. Le premier volume se veut une *connaissance à long terme* décrivant les théories du multimédia. Le deuxième volume est plus appliqué et, donc, plus sujet à se *démoder* avec l'évolution de la technologie et de l'économie de marché. Mais, les modes vont et viennent avec des modifications, des variantes qui ne sont pas toujours radicales. Par exemple, le passage de MPEG2 à MPEG4-AVC est fait d'une multitude d'optimisations qui rend les clips vidéo accessibles depuis un téléphone mobile. Cependant, les concepts et les algorithmes utilisés par MPEG4-AVC restent, dans leurs lignes générales, identiques à ceux de MPEG2.

Plus précisément, le premier volume traite des *théories de la compression*. Il aborde en trois volets les principaux concepts nécessaires à la compréhension de l'ensemble des outils du multimédia décrit dans les deux autres parties.

Le premier volet (chapitre 2, volume 1) est le fondement qui a donné naissance à tous les outils qui suivent. Il survole l'ensemble des *transformées de fonctions*. Ces transformées amènent à considérer une fonction non seulement dans le domaine *temporel* ou *spatial*, mais aussi, dans le domaine *fréquentiel*.

Récemment découvert, un ensemble de transformées, issu de la convergence entre la théorie des signaux et l'analyse mathématique, permet d'observer une fonction dans le domaine *spatiofréquentiel*.

Bien que ces notions de transformées reposent sur des théories mathématiques avancées, nombre de raccourcis sont pris pour ne pas trop s'éloigner de l'objectif initial de ce livre; à savoir, donner une *description intuitive* des outils du multimédia.

Ce volet a été conçu pour apporter le minimum nécessaire à toute personne soucieuse d'approfondir un point de vue théorique précis. Des références sont insérées régulièrement comme des clés pour tel ou tel aspect des tranformées. De plus, des annexes viennent compléter certains points.

Le deuxième volet (chapitre 3, volume 1) étudie tout ce qui concerne (1) la numérisation et (2) la compression des signaux.

La *numérisation*, ou, par anglicisme, la *digitalisation*, est la première brique de l'édifice multimédia. Elle définit les propriétés qu'un *convertisseur* (par exemple, un scanner familial) doit avoir pour transformer un *signal analogique* en un *signal numérique* ou celles qu'un *capteur numérique* (par exemple, un appareil photographique numérique) doit avoir pour faire une capture numérique correcte.

Ces propriétés – et les théorèmes qui en découlent – ont été vus dans le premier volet de cette partie. L'analyse fréquentielle du signal, quelle que soit sa nature, est primordiale puisque le signal numérique n'est plus décrit par des valeurs continues mais par des valeurs discrètes.

Ainsi, un signal sonore qui est continu dans le temps, est décrit par des valeurs régulièrement espacées dans le temps, quand il est numérisé. De même, une image numérique est enregistrée sur une matrice composée d'éléments unitaires appelés *pixels* (raccourci pour *picture elements*).

La deuxième étude de ce volet est la *compression*. Celle-ci vise à diminuer la place prise par un signal (sur un support mémoire ou sur un réseau). Elle peut se faire en éliminant toute information redondante. Dans ce cas, il n'y a pas de différence entre le signal original et le signal reconstruit après décompression. La compression est alors *sans perte*.

Les deux processus de compression, que sont (a) la quantification et (b) la codification, font référence à la *théorie de l'information*, introduite par Claude Shannon. Elle est donc brièvement présentée. Ses liens avec la quantification et la codification sont mis en exergue :

1) la *quantification* est, à la fois, un élément de la numérisation et un élément de la compression :

a) elle est un élément de la numérisation car les amplitudes des signaux continus sont réelles. Et, chacun sait que la représentation des réels sur une machine n'est qu'une approximation des réels mathématiques. Dans certains cas, on peut vouloir des signaux numériques à amplitudes entières. Par exemple, les images au format RGB ont des valeurs de pixels entières, comprises entre 0 et 255. La quantification s'apparente alors à un découpage de l'axe des réels mathématiques en intervalles, tel que chaque

intervalle ait un représentant enregistré sur la machine. Ainsi, à chaque amplitude continue correspond un intervalle. L'amplitude est alors remplacée par le représentant de cet intervalle,

b) la quantification est également un élément de la compression car elle intervient souvent après la numérisation pour augmenter le taux de compression (c'est-à-dire, pour diminuer la taille du signal compressé). Les pertes sont alors importantes et ne doivent pas avoir lieu n'importe où, n'importe quand ;

2) la *codification* récupère le résultat provenant de la quantification du signal numérique. Elle transforme ce signal en un *flux binaire* pour être stocké sur un support adéquat (CD-ROM, DVD, etc.) ou pour être transmis sur un réseau.

Mais, une *compression avec pertes* peut s'avérer obligatoire si la taille engendrée par la compression sans perte n'est pas suffisament faible. On a vu que la quantification est souvent réutilisée pour répondre à ce besoin. Mais, il faut alors contrôler les pertes engendrées.

Dans le troisième et dernier volet de ce premier volume (chapitre 4, volume 1), l'utilisateur est remis au centre du processus afin de contrôler les pertes inévitables si l'on veut transmettre ou stocker un signal.

Ainsi, deux questions fondamentales se posent. Qu'est-ce que l'ouïe ? Qu'est-ce que la vue ? Questions qui ont permis de construire des algorithmes de compression engendrant des pertes qui ne sont pas (trop) génantes pour l'utilisateur.

Ce troisième volet présente, donc, succinctement, les modèles de compression choisis pour les signaux sonores et visuels. Ces modèles autorisent des pertes d'information quand la perception humaine est peu sensible tout en conservant une relativement grande précision quand la perception humaine est plus sensible.

Le deuxième volume présente en deux parties les formats images et les formats audiovisuels du multimédia. La première partie se divise en trois chapitres qui définissent trois modèles de représentation des images.

Le premier chapitre (chapitre 2, volume 2) est dédié aux modèles de compression sans perte des images. Les codeurs décrits servent de référence aux outils de compression avec pertes des images et des audiovisuels.

Le deuxième chapitre (chaitre 3, volume 2) présente le standard grand public actuel (en 2008); à savoir, le format JPEG. Il est d'autant plus incontournable que les premières versions du format MPEG (MPEG1 et MPEG2) le prennent comme base pour la compression vidéo.

Enfin, le troisième et dernier chapitre de cette première partie (chapitre 4, volume 2) décrit le standard JPEG2000. Bien que mis à l'index pour des raisons économiques, il utilise, pour une des premières fois dans le cadre de la compression d'images, nombre d'outils très performants. D'ailleurs, ces outils sont utilisés par la partie vidéo du standard MPEG4.

La deuxième partie de ce volume présente la compression audiovisuelle à travers l'évolution du format MPEG.

Le standard MPEG1 est décrit en premier (chapitre 5, volume 2). Il pose les bases de la compression audio et vidéo.

Puis, le chapitre suivant (chapitre 6, volume 2) présente le standard MPEG2. Celui-ci améliore certains aspects de la compression audio et vidéo proposés par le standard MPEG1. Mais, surtout, il introduit de nouveaux concepts, comme les *profils*, les *niveaux* et la *gestion multimédia* (DSM-CC). Ces concepts permettent une adaptation du format MPEG aux exigences des réseaux. De plus, il autorise une interaction (encore faible) entre l'utilisateur et le produit multimédia.

Le dernier chapitre (chapitre 7, volume 2) aborde le standard MPEG4 et sa version développée (en 2008) MPEG4-AVC.

La compression est révolutionnée par la *modélisation objet* de tout élément audio, visuel, texte, graphique, etc. Ceci reste encore fortement théorique car cette modélisation requiert des outils qui sont encore (en 2008) au stade de la recherche.

Contrairement à ce que son nom laisse entendre, MPEG4-AVC est plus une version optimisée du standard MPEG2 qu'une version du standard MPEG4.

Toutefois, le modèle du standard MPEG4 peut déjà être utilisé dans des situations suffisamment contraintes. Dans ces cas de figure, la modélisation objet permet une forte interactivité où l'utilisateur final peut intervenir au niveau de la présentation du produit multimédia (lors du décodage), mais aussi, au niveau de la conception du produit multimédia (lors du codage). Elle permet également de fusionner en un même produit multimédia des *données synthétiques* (textes, graphiques, sons et images de synthèse, etc.) et des *données naturelles* (photographies, sons et audiovisuels enregistrés, etc.).

Il s'agit là de l'avenir du multimédia. Quels ques soient les outils qui seront développés, l'interaction avec des données naturelles et synthétiques sera de plus en plus forte.

Comment lire ce livre ?

Il est évident que l'on peut lire ce livre dans sa globalité en commençant par le premier volume. Toutefois, j'ai tenté de rendre les volumes et les chapitres les plus indépendants possibles les uns des autres. Quand cela est nécessaire, j'ai inséré des références à des sections de chapitres précédents afin de permettre au lecteur d'approfondir certains aspects.

Par exemple, le chapitre traitant du standard MPEG1 peut être lu séparément des autres. Mais, puisqu'il réutilise les techniques du standard JPEG, celles-ci sont brièvement décrites dans le chapitre MPEG1. Des références au chapitre JPEG du volume 2 permettent d'approfondir la lecture. De même, le chapitre JPEG fait référence au chapitre 2 du volume 1 pour une description plus détaillée de la transformée en cosinus discrète.

Sur le site http://sites.google.com/site/intromm/ vous trouverez les images en couleurs de ce livre ainsi que des compléments d'information.

Chapitre 1

Introduction

Bien que ce volume soit plus mathématique que le deuxième, son objectif n'est pas tant de faire apprécier les mathématiques sousjacentes au multimédia que de fournir des explications concrètes aux fondements de la production et de la diffusion multimédia. Peu de démonstrations sont données dans les chapitres qui suivent. Plutôt, les définitions et les théorèmes y sont expliqués de manière intuitive.

Le lecteur soucieux de poursuivre plus à fond un point théorique pourra le faire grâce aux références bibliographiques régulièrement proposées dans les trois chapitres de cette première partie.

Les bases théoriques du multimédia sont :

– la famille des transformées ;

– la théorie de l'information ;

– la perception humaine.

La *famille des transformées*, présentée au chapitre 2, forme le cœur de la compression des données multimédia. En effet, toute transformation cherche à faire ressortir les caractéristiques pertinentes, fondamentales, de l'*objet* observé. Autrement dit, on cherche à *décorréler* les données concernant cet objet. S'il s'agit d'un signal audio, les mesures faites à un moment donné dépendent plus ou moins des mesures faites dans le passé. Si c'est un signal vidéo, la dépendance est à la fois spatiale et temporelle. Une information est en relation avec les informations situées au même endroit ou autour de cet endroit qui proviennent d'un passé et d'un futur proches. Cette dépendance est appelée *corrélation*.

> *Le terme objet est à prendre au sens large, comme dans le cadre des langages orientés objets. Ainsi un objet est une collection de données partageant certaines propriétés. Par exemple, un objet peut décrire des pixels voisins d'une image qui respecteraient le même critère concernant leurs couleurs. Dans certains cas, l'objet est muni d'outils permettant sa manipulation. Les objets du standard MPEG4 sont définis de cette manière.*

Par exemple, un texte écrit est composé de caractères qui forment des mots. Ces caractères sont donc fortement correllés. Lors de la lecture d'un texte, il est parfois possible de prévoir avec certitude quelle lettre va apparaître à l'instant suivant. De même, pour la parole, un son perçu à un instant t donné, associé à plusieurs sons le précédant dans le temps, permet d'identifier le phonème qui est à son origine.

Les transformées utilisées dans le cadre de la compression audio, image et vidéo sont de type fréquentiel. En effet, un signal naturel – c'est-à-dire, mesuré par un capteur – peut être décrit par les fréquences qui le composent. L'analyse est alors une projection du signal sur une famille de fonctions oscillantes de moyenne nulle. Les valeurs de cette projection caractérisent le signal observé. La *transformée en cosinus discrète* et la *décomposition en ondelettes* sont les deux outils de la compression.

Le chapitre 3 explique la formation des signaux numériques à partir de leurs formes analogiques. Une fois ces signaux numérisés, les techniques de quantification, de codage et de prédiction sont décrites dans le cadre de la *théorie de l'information*.

Enfin, le chapitre 4 présente la nature physique des signaux et la perception que l'humain en a. Ces études permettent d'obtenir des taux de compression élevés, tout en conservant une haute qualité du signal reconstruit après décompression. Ce chapitre se divise en deux parties :

1) le modèle physique et le modèle perceptif de la couleur qui définissent différents espaces de représentation et de mesure de la couleur. Chaque espace répond à certains critères. Les espaces séparant la luminosité de la chrominance sont utilisés pour la compression image et vidéo ;

2) le modèle physique et le modèle perceptif du son. Les effets de masquage dans le domaine temps-fréquence du système auditif humain sont utilisés pour la compression sonore. Suivant l'efficacité et la qualité voulues, les algorithmes prennent plus ou moins en compte les propriétés du masquage.

Chapitre 2

Les transformées

Comme on le verra dans les chapitres suivants – traitant de la compression de l'audio, de l'image et de la vidéo – les transformées sont fondamentales. Elles sont l'une des clés d'une compression efficace en termes de débits sur un réseau ou en possibilités de stockage, tout en veillant à la qualité de restitution de l'audio, de l'image ou de la vidéo. En particulier, les transformées en ondelettes font sûrement partie des outils du multimédia à venir. On ne peut donc pas les ignorer sous le prétexte que les théories mathématiques et physiques sous-jacentes sont complexes.

Aussi, ce chapitre abordera, de manière *intuitive*, les notions et les principes des différentes transformées utilisées dans le multimédia. Autrement dit, beaucoup de propriétés et de caractéristiques ne sont pas démontrées; qu'elles soient du domaine fréquentiel, de l'espace temps-fréquence ou de l'espace déplacement-échelle. Mais, le lecteur voulant vérifier ou approfondir un aspect particulier trouvera, tout au long de ce chapitre, les références nécessaires.

Après une rapide présentation des deux catégories de transformées en section 2.1, la transformée de Fourier des signaux continus monodimensionnels (les signaux sonores, par exemple) est décrite en section 2.2. Puis, la section 2.3 présente cette transformée pour les signaux discrets (c'est-à-dire numériques). L'extension aux cas des signaux 2D est présentée en section 2.4.

Mais, la transformée de Fourier recourant aux nombres complexes n'est pas idéale pour une transformation rapide et efficace d'un signal. Aussi, la transformée en cosinus discrète, qui est sa version à coefficients réels, est-elle préférée. La section 2.5 la présente, en faisant le lien avec la transformée de Fourier. Cette transformation est primordiale puisqu'elle est utilisée par les standards JPEG et MPEG.

Toutefois, la famille des transformées de Fourier – y compris la transformée en cosinus discrète – présente les signaux dans le domaine spectral, en perdant toute information temporelle (pour les signaux sonores), spatiale (pour les images). La section 2.6 étudie comment remédier à cet inconvénient. Il s'en suit qu'une extension de cette famille est nécessaire pour observer simultanément les caractéristiques spectrales et temporelles[1] des signaux. La section 2.7 introduit les transformées en ondelettes des signaux continus qui découlent de cette constatation. Une analogie est faite entre les séries de Fourier et les séries d'ondelettes en section 2.8.

Puis, l'analyse multirésolution (AMR) est décrite en section 2.9. Cette analyse fait converger les études sur les bancs de filtres avec celles des transformées en ondelettes. Cette convergence a apporté des algorithmes d'une complexité d'implémentation réduite avec des temps de calcul abordables. Pour mieux saisir tout l'intérêt des transformées en ondelettes, la section 2.10 propose une étude de cas avec les ondelettes de Haar.

Enfin, avant de conclure, la section 2.11 introduit les concepts de base des transformées en ondelettes dites de seconde génération. Celles-ci sont sujettes à moins de contraintes et offrent des algorithmes simples à implémenter et suffisamment efficaces en temps de calcul pour être utilisés en *temps réel* (voir annexe A.1).

2.1. Quelles transformées ?

On va scinder l'ensemble des transformées en deux catégories : les *transformées spatiales* et les *transformées fréquentielles*. Cependant, toutes ces transformées ont pour but de décrire le signal observé sous un nouveau point de vue afin d'y percevoir plus nettement certaines caractéristiques et propriétés. Elles visent à *décorréler* le signal (voir annexe A.2).

Si le signal n'est pas le résultat de phénomènes indépendants, purement aléatoires, il est alors évident que l'on peut prédire son allure en l'observant. Par exemple, dans un texte, certaines lettres apparaissent plus fréquemment que d'autres suivant la langue utilisée; on peut parfois être certain de la lettre à venir connaissant le *contexte* (voir annexe A.3) défini par les lettres déjà lues. La *théorie de l'information*, développée par Claude Shannon – et introduite au prochain chapitre – fait référence à cette propriété de l'information.

Les mêmes phénomènes apparaissent avec d'autres supports comme, par exemple, l'enregistrement audio d'un discours. Le fait de pouvoir prédire signifie qu'il y a un lien implicatif, une relation temporelle et/ou spatiale, entre les valeurs du signal. On

1. Ou spatiales ou spatio-temporelles suivant le type de signal observé.

parle alors de signal *corrélé*. La *décorrélation* d'un signal a pour but de supprimer ces relations et de mettre en exergue certaines caractéristiques.

2.1.1. *Les transformées spatiales*

Parmi les transformées spatiales, la plus connue est l'*analyse en composantes principales*. Pour cette catégorie de transformées, la décorrélation se traduit par une transformation linéaire. Autrement dit, l'espace d'observation est toujours le même : le temps pour un signal sonore, l'espace bidimensionnel pour les images, etc. La transformée recherche le meilleur repère dans cet espace pour décrire les valeurs du signal.

La figure 2.1 illustre ce principe où le repère, en traits pointillés mixtes, caractérise mieux le nuage de points que ne le fait le repère initial, d'axes horizontal et vertical. La projection des points sur les axes du nouveau repère, donne une meilleure description de la distribution de ces derniers. On le constate aisément avec le rectangle englobant construit à partir des valeurs minimale et maximale des projetés sur les axes du nouveau repère. Ce rectangle est plus compact que le rectangle englobant construit par projection sur les axes de l'ancien repère.

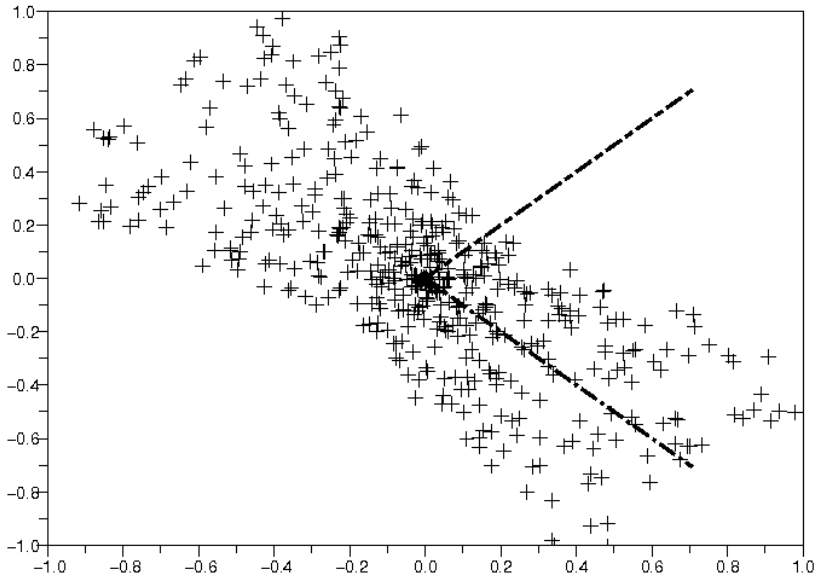


Figure 2.1. *Le principe des transformées en composantes : le repère original subit une rotation qui permet de définir des projections axiales plus pertinentes*

D'une manière générale, il s'agit de trouver le repère dont le premier axe maximise la dispersion au sens des moindres carrés. Ce chapitre ne décrira pas plus ces transformations, car elles ne sont pas directement utilisées en compression.

2.1.2. *Les transformées fréquentielles*

Les transformations fréquentielles sont à l'origine d'avancées formidables dans le domaine de la compression. Dans cette catégorie, l'espace de représentation initial est oublié au profit d'un espace plus adéquat à certaines modélisations mathématiques. Joseph Fourier a été l'initiateur de ce domaine.

Grossièrement, le principe est comparable à celui des transformées géométriques, mis à part que le repère sur lequel on va vouloir projeter notre signal est une famille de fonctions répondant à des critères. Algébriquement, c'est l'espace de Hilbert qui n'est pas abordé dans ce chapitre. Toutefois, le lecteur intéressé pourra se référer à l'excellent livre de Claude Gasquet et Patrick Witomski [GAS 00]. On oublie donc les démonstrations et on fournit uniquement les définitions et les propriétés importantes dans le cadre de la compression.

En premier, la section 2.2 présente la transformée de Fourier car elle est la *mère* de la transformée en cosinus discrète décrite dans la section 2.5. La transformée en cosinus discrète est utilisée par les formats JPEG et MPEG. Ensuite, les sections 2.6 à 2.11 s'intéressent à une famille de transformées découverte et mise en œuvre récemment : les transformées en ondelettes. Elles sont utilisées par le format JPEG2000 et par les derniers formats MPEG. Elles seront sûrement l'outil du cinéma numérique.

2.2. La transformée de Fourier

Bien que, tout au long de ce livre, on s'intéresse aux fonctions *apériodiques discrètes* – c'est-à-dire, aux images, sons et audiovisuels numériques – les *séries de Fourier*, définies pour les fonctions *périodiques continues*, sont étudiées en premier. Les définitions d'une fonction périodique/apériodique et d'une fonction continue/discrète, sont données en annexes A.4 et A.5.

Puis, cette théorie est étendue aux fonctions *apériodiques continues* grâce à la *transformée de Fourier* (FT). Enfin, cette section conclut avec la transformée qui a révolutionné le traitement du signal, à savoir la *transformée de Fourier discrète* (DFT). Comme son nom l'indique, elle permet d'obtenir une représentation spectrale *discrète* d'un signal *discret*.

2.2.1. *Séries de Fourier pour fonctions périodiques*

Initialement, Joseph Fourier a émis la proposition que tout signal périodique peut être décrit par une somme pondérée de fonctions trigonométriques dont les périodes sont des multiples de la *fréquence fondamentale*, F, du signal. Cette somme étant infinie, c'est donc une série appelée *série de Fourier*.

DÉFINITION 2.1.– *Soit un signal $s(t)$ monodimensionnel périodique, de période fondamentale, $T = \frac{1}{F}$. Sa série de Fourier est :*

$$s(t) = \frac{a(0)}{2} + \sum_{n=1}^{\infty} (a(nF)\cos(2\pi nFt) + b(nF)\sin(2\pi nFt)) \qquad (2.1)$$

où les coefficients $a(nF)$ et $b(nF)$ valent :

$$\begin{aligned} a(nF) &= \tfrac{2}{T}\int_0^T s(t)\cos(2\pi nFt)\mathrm{d}t \\ b(nF) &= \tfrac{2}{T}\int_0^T s(t)\sin(2\pi nFt)\mathrm{d}t \end{aligned}$$

Le coefficient $a(0)$, appelé composante continue ou principale (DC[2]), informe sur la valeur moyenne du signal et se calcule suivant l'égalité :

$$a(0) = \frac{2}{T}\int_0^T s(t)\mathrm{d}t = 2\overline{s(t)}$$

Les autres coefficients $a(nF)$ et $b(nF)$ $(n \neq 0)$ sont les harmoniques de la série.

DÉFINITION 2.2.– *La période fondamentale, T, mesurée en secondes (unité s), définit la durée du signal avant que celui-ci ne se répète.*

Son inverse, la fréquence fondamentale, $F = \frac{1}{T}$, mesurée en Hertz (unité Hz), donne alors le nombre de périodes apparaissant dans le signal durant une seconde.

La figure 2.2 illustre ces notions de période et fréquence fondamentale avec deux exemples de signaux périodiques achromatiques; de la forme $\sin(\omega t)$ avec $\omega = 2\pi F$.

2. Le symbole *DC* signifie *direct current* en anglais.

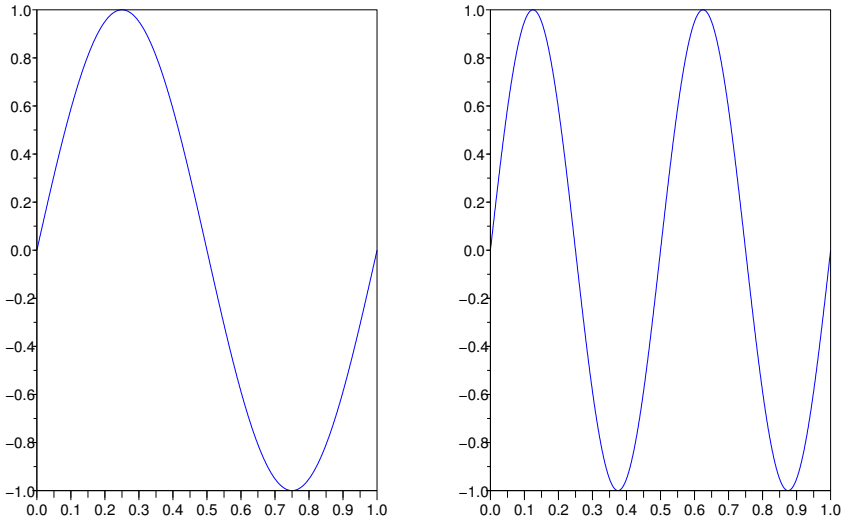


Figure 2.2. *Deux signaux périodiques, dits achromatiques, car chacun reposant sur une et une seule fréquence : sa fréquence fondamentale. Le signal de gauche est de la forme* $\sin(2\pi t)$. *Il est de période et de fréquence égales à* 1*:* $T = 1s$ *et* $F = 1\,\mathrm{Hz}$. *Le signal de droite est de la forme* $\sin(2\pi 2t)$. *Il est de période* $T = \frac{1}{2}s$ *et de fréquence* $F = 2\,\mathrm{Hz}$.

EXEMPLE 2.1.– *La figure 2.3 montre un signal carré périodique* $\left(T = 2\pi; F = \frac{1}{2\pi}\right)$ *:*

$$s(t) = \begin{cases} 1 \textit{ si } 0 \leqslant t < \pi \\ \\ -1 \textit{ si } \pi \leqslant t < 2\pi \end{cases}$$

ainsi que deux de ses développements en séries de Fourier respectivement tronqués à l'ordre 3 *et à l'ordre* 31 *:*

$$S_3(t) = \frac{4}{\pi}\left(\sin(t) + \frac{\sin(3t)}{3}\right)$$

$$S_{31}(t) = \frac{4}{\pi}\sum_{k=0}^{30}\frac{\sin((2k+1)t)}{2k+1}$$

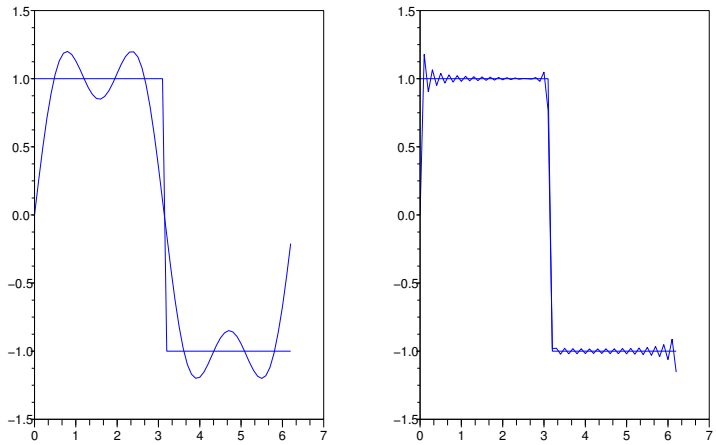


Figure 2.3. *Développement en série de Fourier d'un signal carré périodique* $s(t)$.
La série exhibée par le graphe de gauche est tronquée à l'ordre 3 *:* $S_3(t)$.
Le graphe de droite exhibe la série tronquée à l'ordre 31 *:* $S_{31}(t)$

On peut d'ores et déjà constater :

– que les basses fréquences donnent la forme générale du signal, alors que les hautes fréquences apportent des précisions de détail ;

– que les coefficients, décrivant les amplitudes des sinusoïdes, vont en diminuant, quand les fréquences augmentent; ceci est fortement lié au premier point puisque les hautes fréquences vont permettre de s'approcher de plus en plus finement des points anguleux du signal ;

– qu'en pratique, on ne peut avoir qu'une représentation approximative puisque seul le développement non tronqué correspond exactement au signal original.

2.2.2. *Forme bilatérale complexe*

L'équation (2.1) peut se réécrire sous sa forme *bilatérale complexe* :

$$s(t) = \sum_{n=-\infty}^{+\infty} S(nF) e^{i2\pi nFt}$$

Les fonctions exponentielles complexes, $e^{i2\pi nFt}$, forment une *base orthogonale* de l'espace des signaux. Cette orthogonalité assure une desciption du signal, $s(t)$, en coefficients *indépendants*, $S(nF)$.

REMARQUE 2.1.– *L'annexe A.8 détaille les propriétés des exponentielles complexes.*

Les coefficients, $S(nF)$, sont obtenus par *projection* du signal, s, sur les exponentielles complexes, $e^{i2\pi nFt}$:

$$\begin{aligned} S(nF) &= < e^{i2\pi nFt}, s(t) > \\ &= \frac{1}{T}\int_0^T s(t)e^{-i2\pi nFt}\mathrm{d}t \end{aligned}$$

REMARQUE 2.2.– *La projection est un produit scalaire plus précisément expliqué en annexe A.7.*

Les coefficients, $S(nF)$, et les coefficients, $a(nF)$ et $b(nF)$ de l'équation (2.1) sont en relation :

$$S(nF) = \frac{1}{2}\left(a(nF) - ib(nF)\right) \tag{2.2}$$

Cette représentation introduit des fréquences négatives qui n'ont pas de sens physique pour des signaux réels tels que ceux décrits dans ce livre. Pour s'en convaincre, on peut retrouver les coefficients, $a(nF)$ et $b(nF)$, à partir des coefficients, $S(nF)$.

DÉMONSTRATION 2.1.– *Les propriétés de parité des coefficients $a(nF)$ et $b(nF)$:*

$$\left\{\begin{array}{lcl} a(-nF) &=& a(+nF) \\ b(-nF) &=& -b(+nF) \end{array}\right.$$

permettent d'écrire :

$$\begin{array}{lcl} a(nF) &=& S(nF) + S(-nF) \\ b(nF) &=& i(S(nF) - S(-nF)) \end{array}$$

L'ensemble des valeurs complexes $\{S(nF) \in \mathbb{C}\}_{n\in\mathbb{Z}}$ forme le *spectre en fréquences* du signal.

Le *module* [3] de la fréquence (nF) vaut :

$$|S(nF)| = \frac{1}{2}\sqrt{a(nF)^2 + b(nF)^2}$$

et sa *phase* :

$$\varphi(nF) = \arctan\left(-\frac{b(nF)}{a(nF)}\right)$$

3. Le module du spectre, aussi appelé *amplitude*, correspond à la racine carrée du spectre de puissance.

DÉFINITION 2.3.– *La représentation bilatérale complexe d'un signal, $s(t)$, met en évidence la correspondance unique entre $s(t)$ et $S(nF)$:*

$$\begin{aligned} s(t) &= \sum_{n=-\infty}^{+\infty} S(nF) e^{i2\pi nFt} \\ S(nF) &= \frac{1}{T} \int_0^T s(t) e^{-i2\pi nFt} \mathrm{d}t \end{aligned}$$

Cette réciprocité est notée :

$$s(t) \overset{\mathfrak{F}}{\Longleftrightarrow} S(f)$$

Le spectre S(f) est à valeurs complexes, de partie réelle, $a(f)$, et de partie imaginaire, $b(f)$ (voir équation (2.2)).

REMARQUE 2.3.– *Les propriétés de la représentation bilatérale de la série de Fourier sont fournies en annexe A.9.*

EXEMPLE 2.2.– *La série de Fourier du signal $\cos(2\pi Ft)$ est décrite par le seul coefficient non nul $a_1 = 1$. Ainsi, toutes les valeurs $S(nF)$ valent 0 sauf $S(-F) = S(F) = \frac{1}{2}$. Le spectre vaut $S(f) = \frac{1}{2}(\delta(f+F) + \delta(f-F))$ (voir figure 2.4).*

EXEMPLE 2.3.– *La série de Fourier du signal $\sin(2\pi Ft)$ est décrite par le seul coefficient non nul $b_1 = 1$. Ainsi, toutes les valeurs $S(nF)$ valent 0 sauf $S(-F) = -S(F) = \frac{i}{2}$. Le spectre vaut $S(f) = \frac{i}{2}(\delta(f+F) - \delta(f-F))$ (voir figure 2.5).*

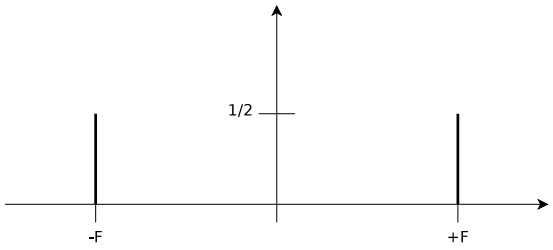


Figure 2.4. *Partie réelle de la représentation spectrale de* $\cos(2\pi Ft)$

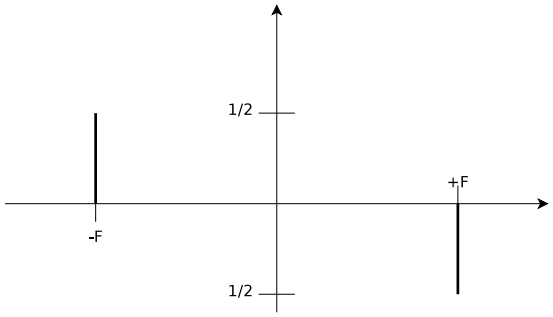


Figure 2.5. *Partie imaginaire de la représentation spectrale de* $\sin(2\pi Ft)$

2.2.3. *Transformée de Fourier pour fonctions apériodiques*

Les fonctions apériodiques peuvent être considérées périodiques en introduisant une période qui tend vers l'infini : $T \to \infty$. La forme bilatérale peut alors être interprétée comme suit :

$$\begin{aligned} s(t) &= \lim_{T\to\infty} \sum_{n=-\infty}^{+\infty} \frac{1}{T} e^{i2\pi \frac{n}{T} t} \left(\int_0^T s(\tau) e^{-i2\pi \frac{n}{T}\tau} \mathrm{d}\tau \right) \\ &= \int_{-\infty}^{+\infty} e^{i2\pi f t} \underbrace{\left(\int_{-\infty}^{+\infty} s(\tau) e^{-i2\pi f \tau} \mathrm{d}\tau \right)}_{\hat{S}(f)} \mathrm{d}f \end{aligned}$$

La fréquence tend vers 0 et le spectre tend vers une fonction continue.

DÉFINITION 2.4.– *La transformée de Fourier (FT) et son inverse s'écrivent :*

$$s(t) = \int_{-\infty}^{+\infty} \hat{S}(f) e^{i2\pi f t} \mathrm{d}f \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{S}(f) = \int_{-\infty}^{+\infty} s(t) e^{-i2\pi f t} \mathrm{d}t$$

De manière analogue à la représentation bilatérale (voir définition 2.3), les parties réelle et imaginaire du spectre valent :

$$\hat{S}_R(f) = \int_{-\infty}^{+\infty} s(t) \cos(2\pi f t) \mathrm{d}t \tag{2.3}$$

$$\hat{S}_I(f) = -\int_{-\infty}^{+\infty} s(t) \sin(2\pi f t) \mathrm{d}t \tag{2.4}$$

L'amplitude et la phase :

$$\begin{aligned} \left|\hat{S}(f)\right| &= \sqrt{\hat{S}_R(f)^2 + \hat{S}_I(f)^2} \\ \varphi(f) &= \arctan\left(\frac{\hat{S}_I(f)}{\hat{S}_R(f)}\right) \end{aligned}$$

REMARQUE 2.4.– *Le spectre de la transformée de Fourier est notée $\hat{S}(f)$ pour le distinguer du spectre des séries de Fourier $S(f)$.*

Certaines conditions d'existence à la transformée de Fourier d'un signal doivent être vérifiées. On les voit sans les justifier. A nouveau, le lecteur voulant approfondir ce sujet peut se référer au livre de Gasquet *et al.* [GAS 00] ou aux premiers chapitres du livre de Stéphane Mallat [MAL 98].

Pour qu'un signal ait une transformée de Fourier inversible, il faut et il suffit que le signal, $s(t)$, et sa transformée, $\hat{S}(f)$, soient d'énergie finie[4] :

$$\int_{-\infty}^{+\infty} s(t)^2 \mathrm{d}t < G_t \text{ et } \int_{-\infty}^{+\infty} \hat{S}(f)^2 \mathrm{d}f < G_f \text{ avec } G_t, G_f \in \mathbb{R}^+$$

De plus, le *théorème de Perceval*)énonce que les énergies, dans le domaine originel et dans le domaine fréquentiel, sont égales. Ainsi, lorsque la transformée de Fourier d'un signal existe, son inverse exact – c'est-à-dire sans perte – existe également ! La réciprocité est donc conservée pour les signaux d'énergie finie.

Heureusement, les événements physiques observés sont d'énergie finie car ceux-ci sont perçus pendant une durée déterminée par des instruments ayant des valeurs de saturation et de déclenchement qui fixent respectivement une valeur maximale et une valeur minimale d'observation.

REMARQUE 2.5.– *Les propriétés de la transformée de Fourier sont données en annexe A.10.*

Le multimédia fait souvent appel aux techniques de traitements du signal. L'opération habituelle consiste à filtrer le signal, x_t, afin d'en conserver la partie intéressante et/ou d'en éliminer les parasites couramment appelés bruits.

REMARQUE 2.6.– *Cette notion de filtrage fait référence aux systèmes linéaires invariants et à la convolution. L'annexe A.11 donne plus de détails sur la convolution.*

4. Aussi appelées fonctions de carrés intégrables ou de carrés sommables.

THÉORÈME DE PLANCHEREL 2.1.– *La convolution du signal $x(t)$ par le filtre linéaire $g(t)$ se note :*

$$x(t) \otimes g(t)$$

Cette technique de convolution, bien qu'indispensable, est fort coûteuse en temps et en mémoire lorsque le signal est observé dans le temps. Mais, si l'on utilise les représentations spectrales, $\hat{X}(f)$ et $\hat{G}(f)$, des signaux, $x(t)$ et $g(t)$, la convolution devient une simple multiplication :

$$x(t) \otimes g(t) \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{X}(f)\hat{G}(f)$$

Le principe est dual : toute convolution dans le domaine fréquentiel revient à faire une simple multiplication dans le domaine temporel.

$$x(t).g(t) \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{X}(f) \otimes \hat{G}(f)$$

Ce théorème est fondamental pour tout traitement d'un signal analogique ou d'un signal numérique. On y fera référence dans la section 2.9, traitant de l'analyse multi-résolution, et dans la section numérisation du prochain chapitre.

2.3. Transformée de Fourier discrète

Puisque, le multimédia repose entièrement sur les signaux numériques, c'est-à-dire, discrets, on introduit, maintenant, la version discrète de la transformée de Fourier.

Un signal $s[t]$ est la version discrète[5] d'un signal $s(t)$ suivant des relevés espacés d'un temps constant T_e, et ceci, pendant une période de temps finie, $T = NT_e$. La fréquence $F_e = \frac{1}{T_e}$ est appelée la *fréquence d'échantillonnage*.

On obtient donc N relevés, $s[kT_e]$, qui, ensemble, forment $s[t]$:

$$s[t] = \sum_{k=0}^{N-1} s[kT_e] = \sum_{k=0}^{N-1} s(kT_e)\delta(t - kT_e)$$

où $\delta(x)$ est la fonction de Dirac qui vaut 1 en $x = 0$ et 0 partout ailleurs.

5. Les crochets sont utilisés pour représenter les signaux discrets, alors que les parenthèses sont associées aux signaux continus.

De manière similaire, le spectre, $\hat{S}[f]$, du signal numérique, $s[t]$, est la version discrète du spectre $S(f)$. Il est décrit par N échantillons spectraux, $\hat{S}[nF_e] = \hat{S}\left[\frac{n}{T_e}\right]$:

$$\begin{aligned}
\hat{S}[f] &= \sum_{n=0}^{N-1} \hat{S}\left[nF_e\right] \\
&= \sum_{n=0}^{N-1} \hat{S}\left(\frac{n}{T_e}\right) \delta\left(f - nF_e\right) \\
&= \sum_{n=0}^{N-1} \left(\frac{1}{NT_e} \sum_{k=0}^{N-1} s[kT_e] e^{-i2\pi n \frac{kT_e}{T}}\right) \delta\left(f - nF_e\right) \\
&= \sum_{n=0}^{N-1} \left(\frac{1}{NT_e} \sum_{k=0}^{N-1} s[kT_e] e^{-i2\pi k \frac{n}{N}}\right) \delta\left(f - nF_e\right)
\end{aligned}$$

Autrement dit, les N échantillons spectraux $\hat{S}[\frac{n}{T_e}]$ sont les approximations, au sens de la formule des trapèzes, des coefficients du cas continu périodique, de période T :

$$S(nF) = \frac{1}{T} \int_0^T s(t) e^{-i2\pi n \frac{t}{T}} \mathrm{d}t$$

La figure 2.6 met en évidence les différentes valeurs utilisées pour décrire $s[t]$, d'une part, et $\hat{S}[f]$, d'autre part.

REMARQUE 2.7.– *La fréquence d'échantillonnage étant constante, elle peut être omise dans les écritures.*

REMARQUE 2.8.– *On constate que le signal, $s[t]$, est entièrement décrit par N échantillons, $s[k]$. De même, sa représentation fréquentielle, $\hat{S}[f]$, est décrite par exactement N échantillons, $\hat{S}[n]$.*

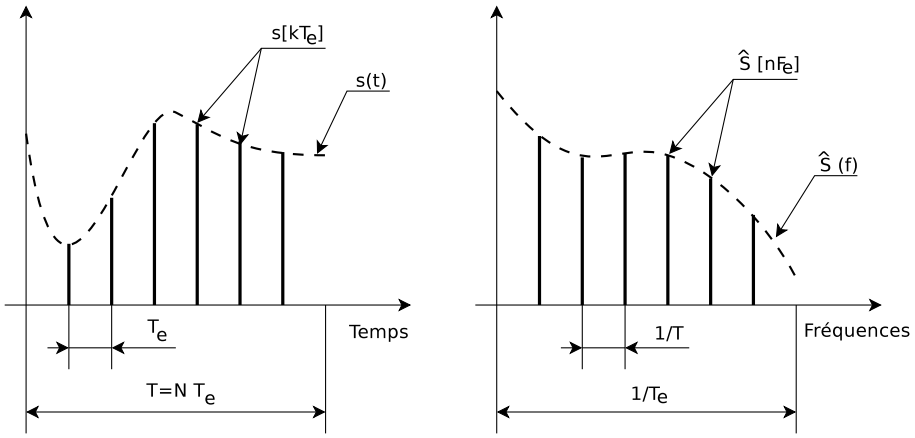


Figure 2.6. *La transformée de Fourier discrète*

REMARQUE 2.9.– *Dans certains cas, la périodicité supposée, $T = NT_e$, peut engendrer une réponse spectrale quelque peu éloignée de la réalité. Le problème est de pouvoir saisir cette réalité. En d'autres termes, il faut définir le nombre, N, d'échantillons nécessaires pour représenter correctement le signal dans sa globalité. Le chapitre suivant revient sur ce point (voir section 3.1).*

DÉFINITION 2.5.– *La transformée de Fourier discrète (DFT) s'écrit :*

$$s[k] = \sum_{n=0}^{N-1} \hat{S}[n] e^{i2\pi n \frac{k}{N}} \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{S}[n] = \frac{1}{N} \sum_{k=0}^{N-1} s[k] e^{-i2\pi n \frac{k}{N}}$$

Les propriétés de la transformée de Fourier sont conservées par sa version discrète.

2.4. Les signaux 2D

La transformation de Fourier se généralise facilement aux signaux 2D, qu'ils soient continus ou discrets. Les décompositions fréquentielles sont associées à chacun des axes. Par exemple, dans le cas d'images 2D (voir figure 2.7), la projection selon l'axe x décrit les *fréquences spatiales verticales* et la projection selon l'axe y décrit les *fréquences spatiales horizontales*. On obtient donc une transformée qui, dans le cas continu, s'écrit :

$$\begin{aligned} \hat{S}(u,v) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} s(x,y) e^{-i2\pi(ux+vy)} \mathrm{d}x\mathrm{d}y \\ s(x,y) &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \hat{S}(u,v) e^{i2\pi(ux+vy)} \mathrm{d}u\mathrm{d}v \end{aligned}$$

REMARQUE 2.10.– *On parle de fréquences spatiales, car elles font référence au nombre d'apparitions visuelles, d'une information donnée.*

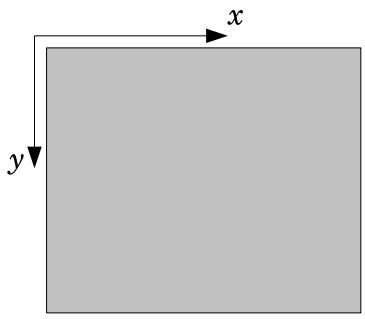


Figure 2.7. *Repère d'une image 2D*

EXEMPLE 2.4.– *Par exemple, si l'on photographie un alignement de poteaux verticaux en se plaçant face à eux, on obtient une photographie similaire à la figure 2.8. Si, ensuite, sans changer de position, on effectue un agrandissment avec l'objectif, la nouvelle photographie (voir figure 2.9) comportera moins de barres verticales et celles-ci apparaitront plus espacées sur l'image. On observe que la fréquence spatiale (le nombre de poteaux) a diminué tandis que la période spatiale (l'espacement des poteaux) a augmenté.*

On retrouve donc les mêmes propriétés de fréquence et de période que pour les signaux temporels.

Dans le cas discret :

$$\begin{aligned} \hat{S}[n,m] &= \frac{1}{NM}\sum_{k=0}^{N-1}\sum_{j=0}^{M-1} s[k,j]e^{-i2\pi\left(n\frac{k}{N}+m\frac{j}{M}\right)} \\ s[k,j] &= \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} \hat{S}[n,m]e^{i2\pi\left(n\frac{k}{N}+m\frac{j}{M}\right)} \end{aligned}$$

avec N et M les nombres d'échantillons suivant l'axe x et l'axe y respectivement. L'annexe A.12 en donne une implémentation possible.

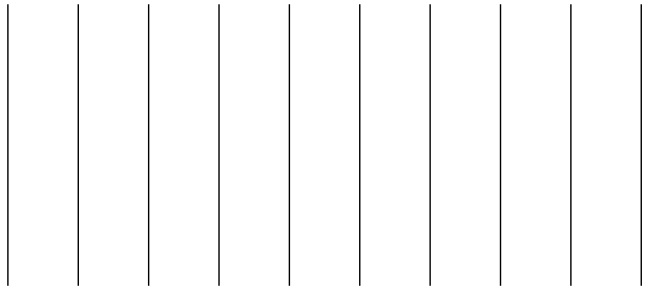


Figure 2.8. *Simulation d'une photographie d'un alignement de poteaux verticaux*

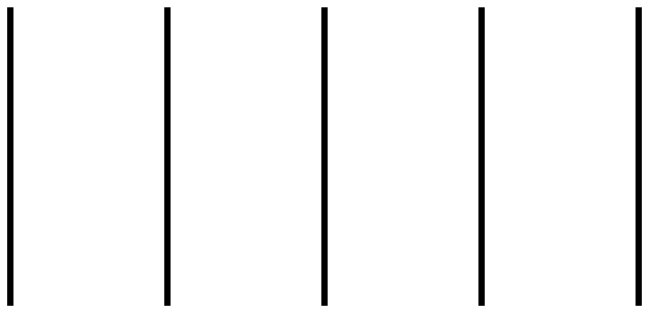


Figure 2.9. *Zoom sur l'alignement de poteaux de la figure 2.8*

EXEMPLE 2.5.– *La figure 2.10a montre la photographie (de taille 512x512 pixels) d'un mandrille. La figure 2.10b montre le spectre de cette image (l'amplitude de la représentation spectrale). Contrairement, au repère haut gauche habituellement choisi (voir figure 2.7), le repère fréquentiel est situé au centre de l'image; la fréquence nulle* $(u = v = 0)$ *est au centre de l'image.*

On constate aisément que les amplitudes vont en diminuant avec l'augmentation des fréquences. Les fortes amplitudes, indiquées par la couleur blanche sur l'image, sont concentrées autour du centre.

La figure 2.10c présente les effets de l'annulation des amplitudes pour les fréquences situées au-delà d'un cercle centré à l'origine et de rayon 10. Ce cercle est visualisé sur la figure 2.10b (le plus petit des deux cercles). La figure 2.10d présente l'image complémentaire à la figure 2.10c. C'est-à-dire, l'image reconstruite en annulant les amplitudes des fréquences à l'intérieur du cercle. La sélection des fréquences est sévère. L'image des hautes fréquences est plus fidèle que celle des basses fréquences. L'image originale (figure 2.10.a) est reconstruite à l'identique en additionnant, pixel à pixel, l'image des basses fréquences (figure 2.10c) et l'image complémentaire (figure 2.10d).

Le procédé est répété avec un cercle de rayon 100. Maintenant, la quasi-totalité de l'énergie de l'image se trouve dans le cercle (le grand cercle de la figure 2.10b). Il s'ensuit que la figure 2.10e est très fidèle à l'image originale. La figure 2.10f présente l'image complémentaire. Elle est faible en intensité et l'information est très localisée :

les hautes fréquences informent sur les détails de l'image observée, alors que les basses fréquences donnent l'allure générale de celle-ci.

A nouveau, l'image originale est reconstruite à l'identique en additionnant l'image de la figure 2.10e avec l'image de la figure 2.10f.

2.5. Transformée en cosinus discrète

Dans les sections précédentes, on a introduit les descriptions fréquentielles des signaux, continus ou discrets, périodiques ou apériodiques. Et, bien qu'il existe des algorithmes efficaces tels que la *transformée de Fourier rapide* (TFR), ces descriptions présentent un inconvénient majeur d'un point de vue purement pratique : elles décrivent un signal réel quelconque par un spectre complexe. Ainsi, tout système voulant utiliser ce type de description fréquentielle est amené à manipuler des nombres complexes qui vont augmenter l'espace mémoire occupé ainsi que les erreurs dues aux arrondis de calcul.

Pour ces raisons, la transformée en cosinus discrète (DCT) est préférée à la transformée de Fourier. L'approche est similaire à celle de la DFT mis à part que la phase

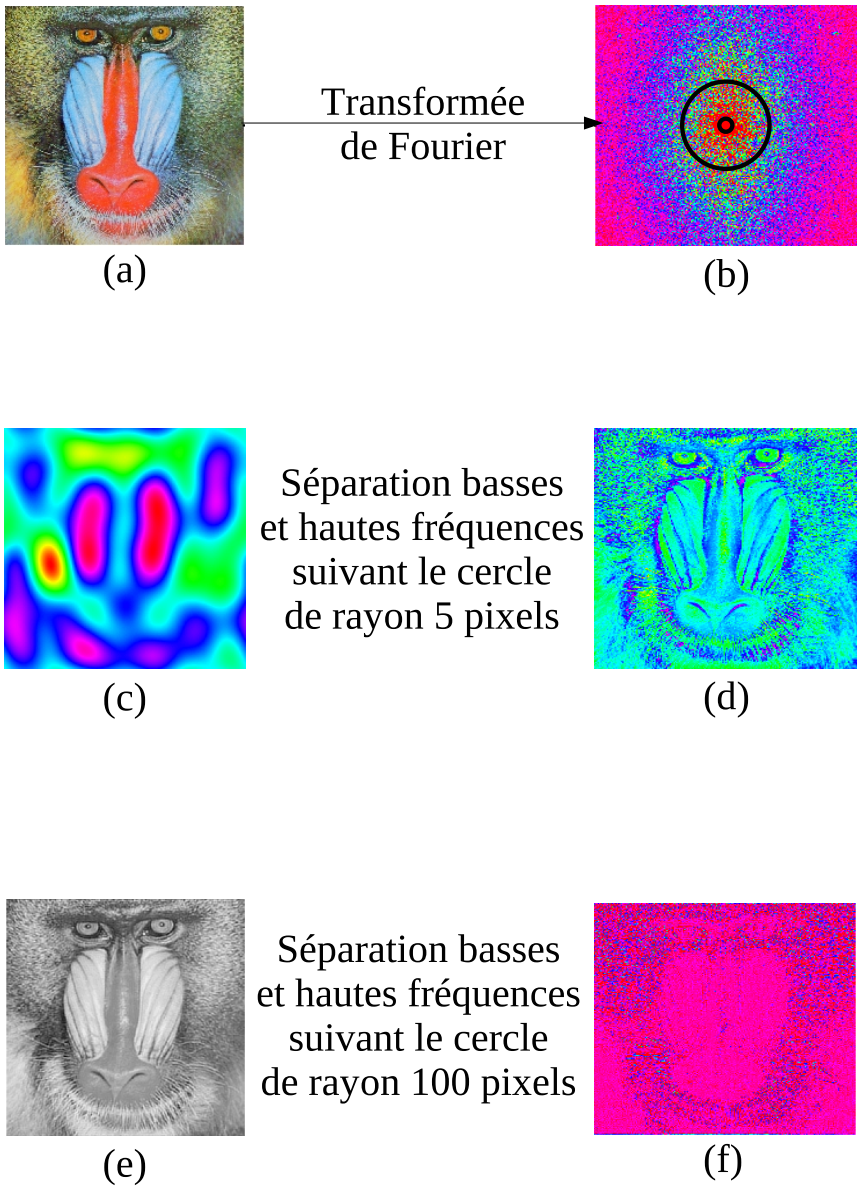


Figure 2.10. *Illustration par l'exemple de la transformée de Fourier discrète : (a) l'image originale d'un mandrille; (b) son spectre; (c) l'image obtenue par transformation inverse des basses fréquences se situant dans le petit cercle de l'image (b); (d) l'image complémentaire à l'image (c) obtenue par transformation inverse des hautes fréquences; (e) l'image de la transformée inverse des basses fréquences se situant dans le grand cercle; (f) l'image complémentaire à l'image (e).*

n'est plus prise en compte. La projection n'est plus une fonction exponentielle complexe, mais, une fonction cosinus qui génère des coefficients réels.

On observe la parenté entre la DFT et la DCT, en symétrisant le signal *réel*, $s[k]$, défini par ses N échantillons (voir figure 2.11) :

$$x[k] = \begin{cases} s[k] & 0 \leqslant k < N \\ s[2N-1-k] & N \leqslant k < 2N \end{cases}$$

En effet, si l'on décrit le signal comme la somme de deux signaux, un réel, $x_R[k]$, et un autre imaginaire, $x_I[k]$:

$$x[k] = x_R[k] + ix_I[k]$$

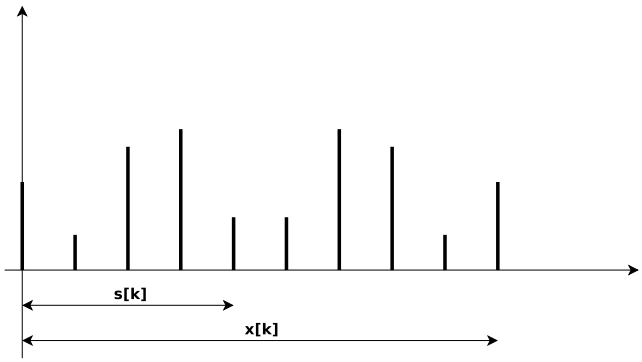


Figure 2.11. *$s[k]$ est la version symétrisée du signal $x[k]$*

La DFT s'écrit alors :

$$\hat{X}[n] = \frac{1}{2N}\left(\hat{X}_R[n] + i\hat{X}_I[n]\right)$$

avec :

$$\begin{aligned} \hat{X}_R[n] &= \sum_{k=0}^{2N-1} x_R[k]\cos\left(2\pi k\tfrac{n}{N}\right) + x_I[k]\sin\left(2\pi k\tfrac{n}{N}\right) \\ \hat{X}_I[n] &= -\sum_{k=0}^{2N-1} x_R[k]\sin\left(2\pi k\tfrac{n}{N}\right) - x_I[k]\cos\left(2\pi k\tfrac{n}{N}\right) \end{aligned}$$

La propriété de parité de la DFT (voir annexe A.10) indique que le spectre fréquentiel d'un signal réel, pair, est également réel et pair. On en déduit alors que

$x_I[k] = \hat{X}_I[n] = 0$ et que la DFT peut être réécrite comme suit :

$$\begin{aligned} \hat{X}[n] &= \frac{1}{2N} \sum_{k=0}^{2N-1} x[k] \cos\left(2\pi k \frac{n}{N}\right) \\ x[k] &= \sum_{n=0}^{2N-1} \hat{X}[n] \cos\left(2\pi k \frac{n}{N}\right) \end{aligned}$$

Cette formulation de la DFT conduit à la version connue de la DCT [PRO 06].

DÉFINITION 2.6.– *La transformée en cosinus discrète s'écrit :*

$$s[k] = \sum_{n=0}^{N-1} \alpha(n) \hat{S}[n] \cos \frac{(2k+1)n\pi}{2N} \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{S}[n] = \alpha(n) \sum_{k=0}^{N-1} s[k] \cos \frac{(2k+1)n\pi}{2N}$$

avec :

$$\alpha(0) = \sqrt{\frac{1}{N}} \textit{ et } \alpha(n) = \sqrt{\frac{2}{N}} \textit{ pour } 1 \leqslant n < N$$

Comme avec la TFD, la DCT se généralise à plusieurs dimensions. Pour les images numériques 2D, la DCT s'écrit :

$$\begin{aligned} \hat{S}[n,m] &= \frac{2\alpha(n)\alpha(m)}{\sqrt{NM}} \sum_{k=0}^{N-1} \sum_{j=0}^{M-1} s[k,j] \cos \frac{(2k+1)n\pi}{2N} \cos \frac{(2j+1)m\pi}{2M} \\ s[k,j] &= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \frac{2\alpha(n)\alpha(m)}{\sqrt{NM}} \hat{S}[n,m] \cos \frac{(2k+1)n\pi}{2N} \cos \frac{(2j+1)m\pi}{2M} \end{aligned}$$

avec :

$$\alpha(0) = \frac{\sqrt{2}}{2} \text{ et } \alpha(n) = 1 \text{ pour } n \neq 0$$

Dans les algorithmes de compression des prochains chapitres, la DCT est appliquée sur des blocs d'images (de taille 8×8, 16×16, etc.). Aussi, les exemples qui suivent appliquent la DCT sur des signaux discrets 1D constitués de huit échantillons.

EXEMPLE 2.6.– *Soit un signal numérique constitué de huit échantillons de valeur 1, montré sur le graphe de gauche de la figure 2.12 :*

$$s[k] = 1(0 \leqslant k < 8)$$

Le graphe de droite exhibe le résultat de l'application de la DCT. Puisque tous les échantillons sont égaux à 1, la valeur moyenne suffit à le décrire. C'est ce que l'on constate sur le graphique de la DCT où la fréquence nulle prend pour valeur la moyenne pondérée du signal et où toutes les autres fréquences sont à 0.

EXEMPLE 2.7.– *Soit le signal carré, déjà étudié au début de ce chapitre, valant* 1 *pour les échantillons indicés de* 0 *à* $3(< \pi)$ *et* -1 *pour les échantillons indicés entre* 4 *et* 7 *(voir figure 2.3). On a observé que la série de Fourier tronquée à l'ordre* 31

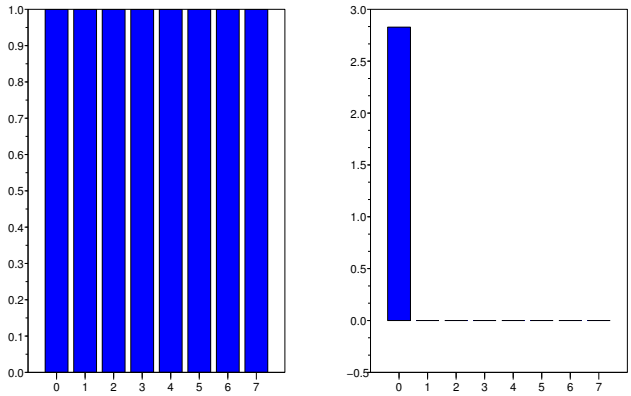


Figure 2.12. *Le graphe de gauche montre un signal carré :* $s[k] = 1(0 \leqslant k < 8)$*, celui de droite sa DCT 1D*

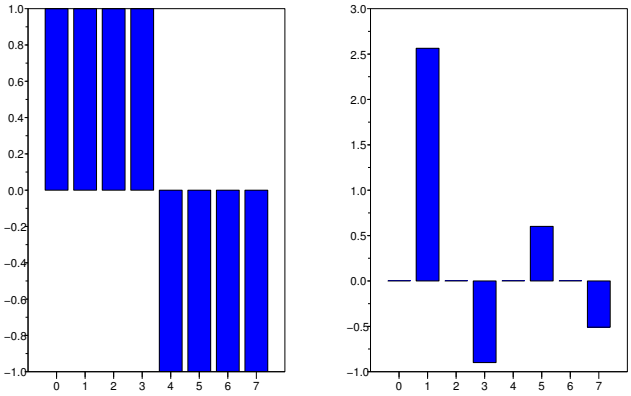


Figure 2.13. *Le graphe de droite montre la DCT du signal carré du graphe de gauche*

décrivant ce signal est de la forme :

$$S_{31}(t) = \frac{4}{\pi} \sum_{k=0}^{30} \frac{\sin((2k+1)t)}{2k+1}$$

On retrouve ce résultat avec la DCT du signal. Dans la figure 2.13, les fréquences paires sont à 0 *et les valeurs des fréquences impaires décroissent en même temps que la fréquence augmente.*

EXEMPLE 2.8.– *Soit l'image du mandrille, dont l'analyse fréquentielle – à l'aide d'une transformée de Fourier discrète – a déjà été étudiée dans l'exemple 2.5 (page 34).*

La figure 2.14a montre l'image originale. La figure 2.14b exhibe la représentation spectrale de la transformée en cosinus discrète. Le repère est situé en haut à gauche. Les basses fréquences sont d'amplitudes plus fortes que les amplitudes des hautes fréquences.

La figure 2.14c montre l'image du mandrille quand seules sont conservées les dix plus basses fréquences. La figure 2.14d montre l'image complémentaire obtenue par transformée inverse en annulant les amplitudes des dix plus basses fréquences. D'un point de vue psychovisuel, l'image des hautes fréquences est plus fidèle que l'image des basses fréquences.

L'image originale de la figure 2.14a est reconstruite à l'identique est additionnant, pixel à pixel, l'image des basses fréquences (figure 2.14c) avec l'image des hautes fréquences (figure 2.14d).

Le procédé est répété, mais, en conservant les cent plus basses fréquences. Les figures 2.14e et 2.14f sont les résultats obtenus, respectivement, à partir des basses fréquences et des hautes fréquences. L'approximation, fournie par l'image des basses fréquences, est, maintenant, bien plus fidèle à l'original. Comme avec la transformée de Fourier, l'image des hautes fréquences montre les détails supprimés de l'image d'approximation. Ces détails sont très localisés. En ajoutant l'image de la figure 2.14f à l'image d'approximation, l'image originale est reconstruite à l'identique.

En conclusion, la DCT se comporte de manière analogue à la transformée de Fourier. Son avantage réside dans l'utilisation de coefficients réels et non complexes.

2.6. Localisation de l'information

Avec la DCT, on a maintenant un algorithme et des outils efficaces en temps de calcul et en mémoire. Toutefois, il est à noter que la caractéristique principale des

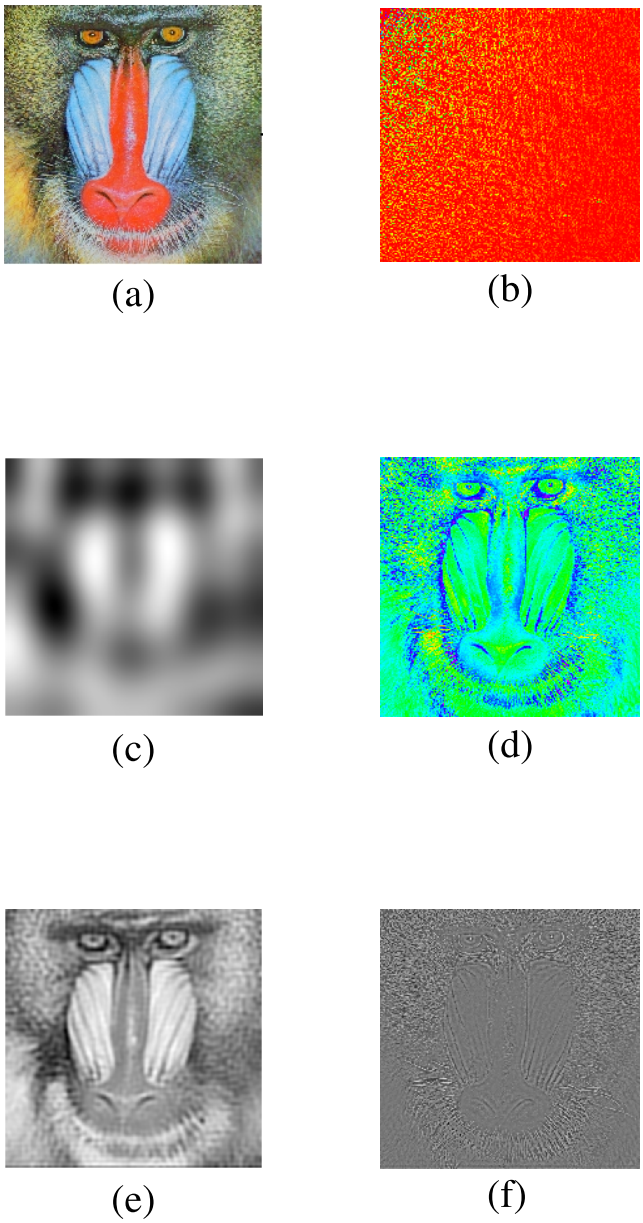


Figure 2.14. *Illustration par l'exemple de la transformée en cosinus discrète : (a) l'image originale d'un mandrille; (b) son spectre; (c) l'image obtenue par transformation inverse en ne conservant que les dix plus basses fréquences; (d) l'image complémentaire à l'image (c) obtenue par transformation inverse des hautes fréquences; (e) l'image de la transformée inverse des cent plus basses fréquences; (f) l'image complémentaire à l'image (e).*

transformées étudiées jusque là est en même temps une limite à la précision de la description fréquentielle.

En effet, la totalité des échantillons du signal est prise en compte pour chaque coefficient fréquentiel. Ainsi, lors d'une étude fréquentielle d'un signal, on perd toute information de position – temporelle ou spatiale – qui permettrait d'identifier les lieux où apparaissent certaines fréquences.

Soient deux exemples de signaux temporels, $s_1(t)$ et $s_2(t)$, tels qu'ils soient tous deux composés de trois sinusoïdes de fréquences respectives 10 Hz, 25 Hz et 50 Hz. Bien que partageant la même base sinusoïdale, ces signaux peuvent être fondamentalement différents (voir figures 2.15 et 2.16) :

$$s_1(t) = \cos((2\pi)10t) + \cos((2\pi)25t) + \cos((2\pi)50t)$$

$$s_2(t) = \begin{cases} \cos((2\pi)10t) & \textbf{si } 0 \leqslant t < 0.3 \\ \cos((2\pi)25t) & \textbf{si } 0.3 \leqslant t < 0.6 \\ \cos((2\pi)50t) & \textbf{\textit{si}} \ 0.6 \leqslant t < 1 \end{cases}$$

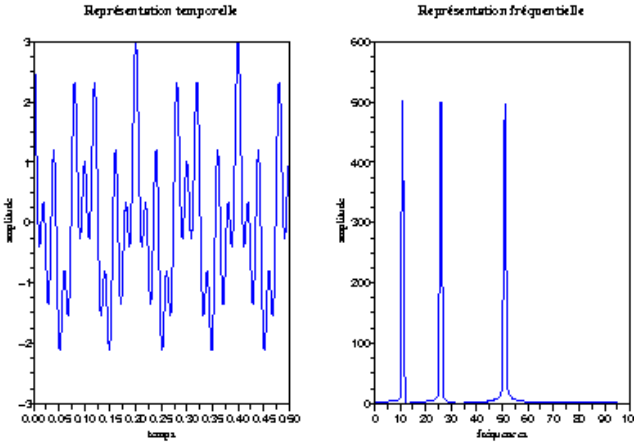


Figure 2.15. *Les représentations temporelle et spectrale du signal* $s_1(t)$

Leurs représentations fréquentielles sont très similaires[6] : dans les deux cas, trois pics, significatifs des fréquences définissant ces deux signaux, apparaissent. Mais,

6. Les faibles amplitudes au sein du spectre de la figure 2.16 soulignent les points singuliers marquant le passage d'une fréquence à la suivante.

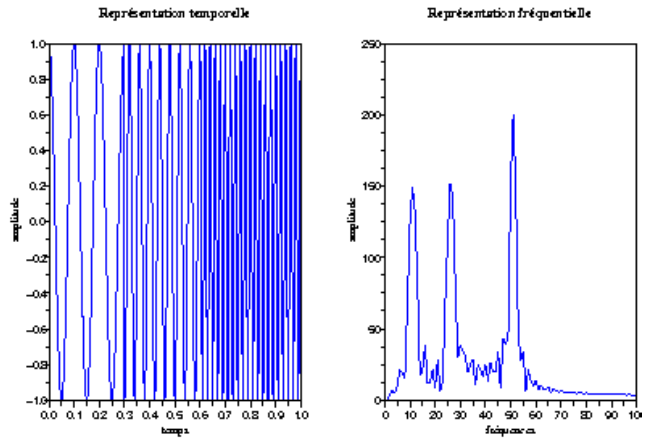


Figure 2.16. *Les représentations temporelle et spectrale du signal $s_2(t)$*

leurs réprésentations temporelles sont clairement différentes. Pour le signal $s_1(t)$, les trois sinusoïdes interviennent à tout moment. Le spectre décrit alors pleinement ce signal. En revanche, pour le signal $s_2(t)$, chaque sinusoïde a sa propre plage temporelle. Dans ce cas, le spectre indique toujours la présence des trois sinusoïdes, mais, il est incapable d'indiquer leurs plages temporelles.

La solution immédiatement envisageable est de recourir à un fenêtrage du signal observé lors de l'application de la transformée souhaitée (la DCT par exemple).

EXEMPLE 2.9.– *Maintenant, on découpe l'image du mandrille en blocs contigus, de taille 8x8. Ensuite on applique la DCT à chacun des blocs. Puis, à partir des blocs spectraux, on reconstruit deux images. Une des images est obtenue par transformation inverse en ne conservant que les deux plus basses fréquences de chaque bloc spectral (voir figure 2.17b), tandis que l'autre image est obtenue en annulant les deux plus basses fréquences de chaque bloc (voir figure 2.17c).*

Le fait d'avoir découpé l'image en blocs a permis de fortement diminuer les fréquences conservées (deux fréquences sur 64 pour chaque bloc de taille 8x8), sans que l'image soit illisible. C'est un gain important si l'on se réfère à l'exemple 2.8.

Toutefois, on a sévèrement restreint les fréquences dans l'image 2.17b et les frontières entre les blocs apparaissent. Cet effet indésirable peut être diminué en conservant plus de basses fréquences pour la reconstruction de l'image 2.17b. Ce procédé relativement simple est celui utilisé par le standard JPEG.

Tout comme en photographie, ce fenêtrage joue le rôle d'obturateur que l'on fait glisser tout au long du signal. A chaque nouvelle position de la fenêtre, la transformée

est appliquée. Ainsi, elle fournit une description fréquentielle en chacune des positions de la fenêtre.

Ce fenêtrage revient à pondérer le signal par une fonction à support fini ou qui décroît très vite, dès que l'on s'éloigne du centre de la fenêtre. Le plus connu et le plus adéquat des obturateurs est celui introduit par Gabor [LEE 96] : il propose d'utiliser une fonction exponentielle pour pondérer les valeurs du signal. L'obturateur prend alors la forme d'une gaussienne qui décroît fortement en sa périphérie. Dans tous les cas, quel que soit l'obturateur utilisé, on obtient des descriptions *temps-fréquence* des signaux 1D et des descriptions *espace-fréquence* des signaux 2D.

En notant ω l'obturateur, la transformée de Fourier fenêtrée 1D (STFT) s'écrit :

$$\hat{S}^{\omega}(t_0, f) = \int_{-\infty}^{+\infty} \underbrace{(s(t)\omega^{\star}(t - t_0))}_{\text{fenêtrage}} e^{-i2\pi ft} \mathrm{d}t \tag{2.5}$$

Le fenêtrage est effectué par un *produit scalaire* du signal $s(t)$ avec l'obturateur $\omega(t - t_0)$ (voir annexe A.7). On constate que l'écriture est bien celle de la transformée de Fourier mis à part que l'obturateur vient restreindre la zone d'observation du signal au voisinage du temps t_0. L'effet d'annulation du signal en dehors de la fenêtrage centrée en t_0 est illustré en figure 2.18.

La taille minimale de ce fenêtrage est définie par le *principe d'incertitude de Heisenberg*. Celui-ci stipule que toute précision dans l'observation suivant un axe se fait aux dépends des autres axes [MAL 98]. Autrement dit, pour un signal 1D, les précisions en temps et en fréquence ont une limite en deçà de laquelle il n'est plus possible d'observer les deux aspects simultanément.

Par exemple, la transformée de Fourier fournit une représentation exacte en fréquences, mais, elle perd toute information temporelle. Inversement, le signal dans sa forme originale indique les informations avec précision en temps, au détriment de l'aspect fréquentiel.

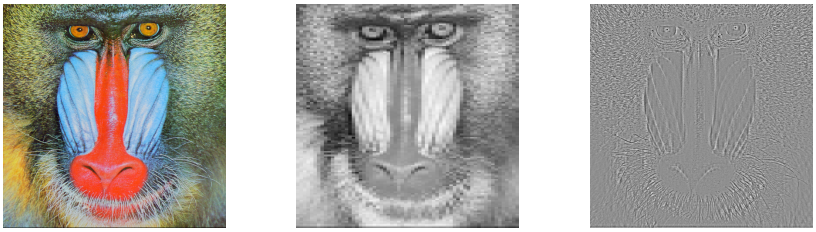


Figure 2.17. *Les effets de l'application de la DCT sur des blocs de taille 8x8 : (a) l'image originale du mandrille; (b) l'image reconstruite en ne conservant que les deux plus basses fréquences de chaque bloc; (c) l'image reconstruire complémentaire à l'image (b)*

Dans le plan temps-fréquence, le principe d'incertitude se traduit par l'inégalité :

$$\sigma_t \sigma_f \geqslant \frac{1}{2}$$

où σ_t définit la largeur de la fenêtre et σ_f celle de la fonction de projection.

Si σ_t est constante et indépendante de la position de la fenêtre ω, la précision suivant l'axe fréquentiel est bornée inférieurement. On a alors un pavage uniforme de l'espace temps-fréquence comme le montre la figure 2.19.

Pour certaines applications, les intervalles de temps des différentes fréquences sont suffisamment bien définis pour avoir une estimation de la taille, σ_t, de la fenêtre ω.

En revanche, d'autres applications ne permettent pas d'associer une plage temporelle à chaque fréquence. Dans ce cas, l'estimation de la taille de l'obturateur est délicate. L'idéal est alors de construire, à partir d'un obturateur générique ω, une famille, $\{\omega_k\}$, d'obturateurs de différentes tailles et de choisir celui qui est le mieux adapté pour chaque intervalle. Les signaux sonores, les photographies et les audiovisuels rentrent dans cette seconde catégorie de signaux.

Le besoin de paramétrer la taille de l'obturateur s'explique aussi par le fait que les hautes fréquences sont très souvent observables sur une durée de temps plus courte que les basses fréquences. Ainsi, pour une même durée d'observation, si le spectre des basses fréquences est correctement observé, celui des hautes fréquences subit des

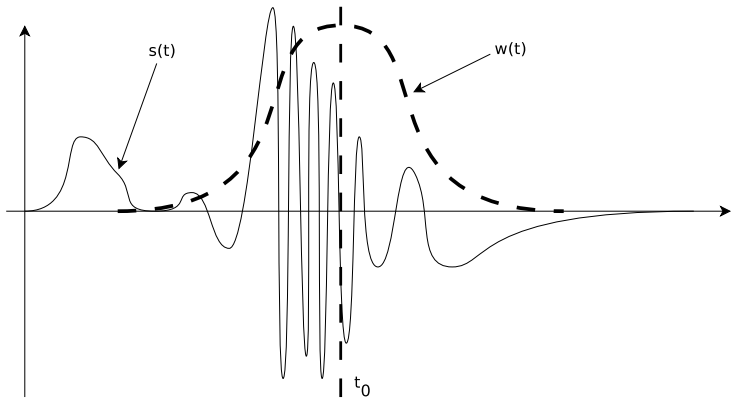


Figure 2.18. *Transformée de Fourier fenêtrée : le produit du signal $s(t)$ avec la fenêtre $\omega(t)$ annule le signal en dehors de la fenêtre d'observation. En $t = t_0$, le produit scalaire $< s(t), \omega(t - t_0) >$ correspond à la somme des valeurs du signal $s(t)$ pondérées par les coefficients définis par les valeurs de l'obturateur $w(t)$ dont le centre de symétrie est déplacé en t_0.*

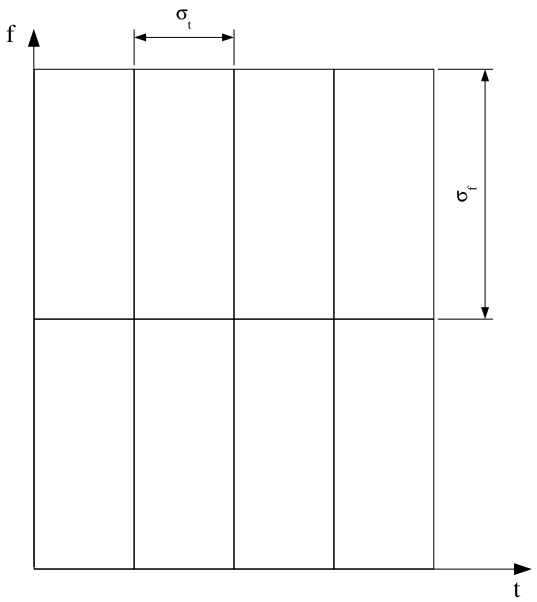


Figure 2.19. *Pavage uniforme de l'espace tempo-fréquentiel*

artéfacts dus à une trop longue période de temps d'observation. Il est donc préférable d'adapter la taille de la fenêtre avec la hauteur des fréquences observées.

La figure 2.20 illustre ce principe : les aires des pavés sont identiques, seules leurs formes changent. Les basses fréquences sont observées sur une grande période de temps, alors que les hautes fréquences ont une plage temporelle plus restreinte. Suivant le principe d'incertitude, le phénomène est observable à l'inverse sur l'axe des fréquences.

Les transformées en ondelettes sont connues pour respecter ce pavage adaptatif. Ce qui fait leur succès pour l'analyse et la compression des signaux sonores et visuels.

2.7. Transformée en ondelettes continue

Le terme d'*ondelette* est utilisé pour signifier qu'il s'agit d'une *onde* tendant très vite vers 0 dès que les valeurs en abscisse s'éloignent de son origine.

D'une certaine manière, elle peut être vue comme une extension de la transformée de Fourier fenêtrée puisque le signal est également fenêtré par une fonction $\psi(t)$. En revanche, la base des exponentielles complexes n'est plus utilisée pour effectuer les projections. La fonction $\psi(t)$ enferme donc en son sein un support fini compact –

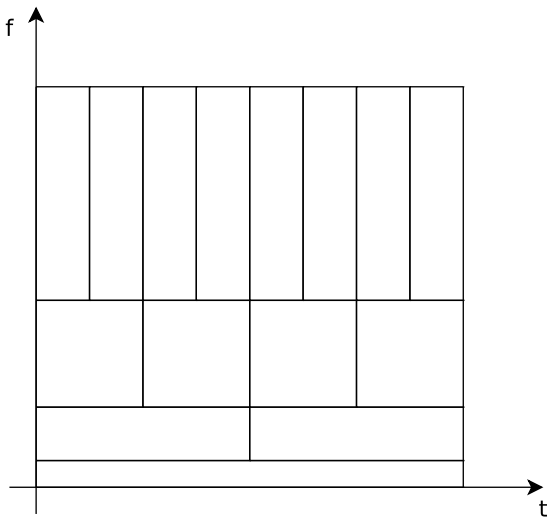


Figure 2.20. *Pavage adaptatif respectant le principe d'incertitude*

comme celui de l'obturateur de la STFT – et une base de projection servant à décrire le signal. Pour assurer la projection, la fonction $\psi(t)$ doit alors être oscillante avec une moyenne nulle afin de conserver l'énergie initiale du signal $s(t)$:

$$\int_{-\infty}^{+\infty} \psi(t)\mathrm{d}t = 0$$

Cette définition de l'ondelette ne décrit pas une fonction précise mais caractérise un ensemble de fonctions. Aussi, on les représente par ψ de façon à être général.

2.7.1. *L'analyse*

L'analyse consiste à décomposer le signal en entrée pour en extraire les informations pertinentes. La définition qui suit, précise cet objectif, en termes de transformée en ondelettes.

DÉFINITION 2.7.– *L'analyse par transformée en ondelettes continue (CWT) fournit un ensemble de coefficients, $\gamma^{(s)}(d,e)$, décrivant le signal $s(t)$ par projection scalaire sur l'ondelette analysante $\psi_{d,e}(t)$:*

$$\gamma^{(s)}(d,e) =< s(t), \psi_{d,e}(t) >= \int_{-\infty}^{+\infty} s(t).\psi^{\star}_{d,e}(t)\mathrm{d}t \qquad (2.6)$$

Pour être plus précis, il ne s'agit pas d'une fonction, mais, d'une famille de fonctions, dont ψ est la génératrice. ψ est appelée l'*ondelette mère* et ses *filles*, $\psi_{d,e}$, sont des versions modifiées par les paramètres d et e :

$$\psi_{d,e}(t) = \frac{1}{\sqrt{|e|}}\psi\left(\frac{t-d}{e}\right)$$

Le paramètre d indique un *facteur de déplacement* de l'ondelette mère. Ce facteur est similaire à la translation de l'obturateur de la tranformée de Fourier fenêtrée. Le paramètre e, appelé le *facteur d'échelle*, est introduit pour générer une version comprimée ($e < 1$) ou dilatée ($e > 1$) de l'ondelette mère. Il est inversement proportionnel à la fréquence :

$$e \propto \frac{1}{f}$$

Ainsi, plus la fréquence augmente, plus le facteur d'échelle diminue et réciproquement. Ce facteur d'échelle joue donc le même rôle que celui des cartes géographiques : là où une carte à faible échelle (par exemple, au 1:25000) permet de voir et de situer des détails, une carte routière, au 1:200000, ne pourra pas fournir une description aussi détaillée.

Autrement dit, le facteur d'échelle définit la *résolution* à laquelle le signal est observé : plus le facteur d'échelle est faible, plus la résolution et la précision en fréquence sont fortes. Inversement, plus le facteur d'échelle augmente, plus la résolution et la précision en fréquence diminuent.

Ces deux paramètres, d et e, fournissent un pavage adaptatif de l'espace *déplacement-échelle*. On remarque, à nouveau, que les transformées en ondelettes fournissent des descriptions dans le plan déplacement-échelle et non plus dans le plan temps-fréquence, comme avec la transformée de Fourier fenêtrée.

Pour les images (signaux 2D), on travaille dans le domaine 3D spatial-échelle : deux axes pour les positions sur l'image, avec des déplacements (d_x et d_y) définis sur chacun des axes, et un axe pour l'échelle.

L'algorithme théorique qui en découle est alors assez simple. Pour un signal 1D, $s(t)$, l'ondelette mère parcourt tout le signal. A chaque déplacement, la projection du signal, s, sur l'ondelette, ψ, fournit le coefficient $\gamma_{d,1}^{(s)}$.

Puis, l'algorithme passe à l'échelle suivante, e, et calcule les nouveaux coefficients $\gamma_{d,e}^{(s)}$ à l'aide de l'ondelette dilatée (et déplacée le long du signal) $\psi_{d,e}(t)$.

$\gamma^{(s)} = \text{CWT}(s, E)$
1 *// cet algorithme est théorique*
2 *// s est le signal 1D en entrée*
3 *// E est le niveau maximal de décomposition*
4 **pour** $e \leftarrow 1$ **à** E
5 **faire** *// le pas est à préciser*
6 **pour** $d \leftarrow -\infty$ **à** $+\infty$
7 **faire** *// le pas est à préciser*
8 $\gamma^{(s)}(d, e) \leftarrow < s(t), \psi_{d,e}(t) >$ *// projeter $s(t)$ sur $\psi_{d,e}(t)$*

EXEMPLE 2.10.– *La dérivée seconde de la gaussienne est une ondelette mère :*

$$\psi(t) = \frac{1}{\sigma^3\sqrt{2\pi}} \left(\frac{t^2}{\sigma^2} - 1 \right) e^{-t^2/2\sigma^2} \tag{2.7}$$

La figure 2.21 montre l'ondelette mère, ψ, et certaines de ses filles. L'ondelette $\psi_{3,2}$ correspond à l'ondelette mère déplacée d'un facteur 3 et dilatée d'un facteur 2 :

$$\psi_{3,2}(t) = \frac{1}{2\sigma^3\sqrt{\pi}} \left(\frac{(t-3)^2}{4\sigma^2} - 1 \right) e^{-\frac{(t-3)^2}{8\sigma^2}}$$

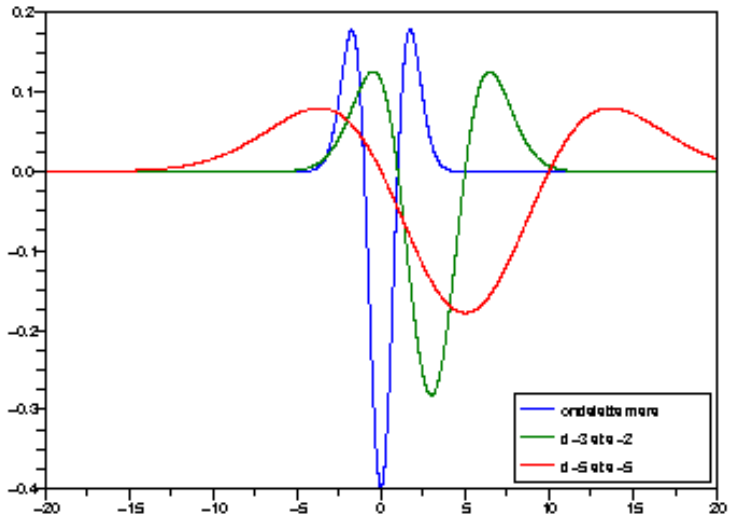


Figure 2.21. *Les ondelettes de Gauss*

2.7.1.1. *Les signaux de dimensions supérieures*

La CWT se généralise aux signaux de dimensions supérieures et en particulier au cas 2D :

$$\begin{aligned}\gamma^{(s)}(m,n,e) &= < s(a,b), \psi_{m,n,e}(a,b) > \\ &= \tfrac{1}{|e|}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} s(a,b).\psi^{\star}\left(\tfrac{a-m}{e},\tfrac{b-n}{e}\right)\mathrm{d}a\mathrm{d}b\end{aligned}$$

La propriété de séparabilité est souvent utilisée. La CWT est appliquée suivant une des deux dimensions du signal, puis, elle est appliquée suivant l'autre dimension.

REMARQUE 2.11.– *Dans l'écriture de l'équation (2.6) de la transformée en ondelette, les facteurs d et e sont continus. En pratique, évidemment, il faut introduire des pas Δd et Δe de discrétisation. Cette question reste en suspend pour l'instant. On y reviendra plus tard.*

2.7.2. *La synthèse*

DÉFINITION 2.8.– *Sachant qu'il y a conservation de l'énergie :*

$$\int_{-\infty}^{+\infty}|s(t)|^2\mathrm{d}t = \frac{1}{C_\psi}\int_0^{+\infty}\int_{-\infty}^{+\infty}|\gamma^{(s)}(d,e)|^2\mathrm{d}d\frac{\mathrm{d}e}{e^2}$$

La synthèse (ou, reconstruction exacte) par transformée en ondelettes continue :

$$s(t) = \frac{1}{C_\psi}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\gamma^{(s)}(d,e)\psi_{d,e}(t)\mathrm{d}d\frac{\mathrm{d}e}{e^2} \tag{2.8}$$

est possible si la condition d'admisibilité est respectée [MAL 98] :

$$C_\psi = \int_{-\infty}^{+\infty}\frac{|\hat{\psi}(\omega)|^2}{\omega}\mathrm{d}\omega < +\infty$$

où $\hat{\psi}(\omega)$ est la transformée de Fourier de $\psi(t)$.

REMARQUE 2.12.– *La condition d'admissibilité revient à s'assurer qu'il y a conservation de l'énergie entre le domaine temporel et le domaine déplacement-échelle. On a déjà eu à vérifier cette propriété pour la transformée de Fourier : théorème de Perceval (voir théorème 2.1 page 30).*

Les bases orthogonales :

$$< \psi_{d_i,e_i}, \psi_{d_j,e_j} >= \delta(d_i-d_j)\delta(e_i-e_j) = \begin{cases} 1 & \textbf{si } d_i = d_j \textbf{ et } e_i = e_j \\ 0 & \textbf{sinon}\end{cases}$$

respectant la condition d'admissibilité, autorisent une reconstruction du signal avec *la même famille d'ondelettes* que celles qui a servi à l'analyse. Ceci est illustré par l'exemple de la section 2.10 qui utilise la base d'ondelettes de Haar.

Mais, fournir une base orthogonale est une contrainte forte. Les *ondelettes biorthogonales* et les *trames d'ondelettes* permettent d'utiliser des familles d'ondelettes non orthogonales qui respectent la condition d'admissibilité. La famille d'ondelettes de synthèse est alors différente de la famille d'ondelettes d'analyse. Elles sont, toutefois, en relation.

Par la suite, ces notions mathématiques de bases biorthogonales ou de trames d'ondelettes ne sont plus abordées. On se contente de poursuivre la présentation des transformées en ondelettes orthogonales.

2.8. Séries d'ondelettes

Dans le cas d'une analyse qui ne serait pas suivie d'une reconstruction par synthèse, les pas de discrétisation pour les paramètres de déplacement, d, et d'échelle, e, peuvent être quelconque.

Lorsque la reconstruction fait partie des objectifs, il existe une relation liant les pas de discrétisation de l'axe des déplacements entre deux échelles. Le fait d'échantillonner, à l'échelle e_1, l'axe des déplacements d à une fréquence F_1 signifie qu'à l'échelle e_2, avec $e_2 > e_1$, la fréquence d'échantillonnage F_2 peut être inférieure à F_1.

Cette relation entre échelles est précisée par le principe de Fourier qui stipule que toute contraction, sur l'axe du temps, équivaut à une dilatation et à un déplacement sur l'axe des fréquences :

$$s(at) \overset{\mathfrak{F}}{\Longleftrightarrow} \frac{1}{a}\hat{S}\left(\frac{f}{a}\right) \tag{2.9}$$

Plus précisément, la relation entre les fréquences d'échantillonnage de deux échelles, e_1 et e_2, est définie par :

$$e_1 F_1 = e_2 F_2 \tag{2.10}$$

On a donc une discrétisation de l'axe d en fonction du rapport des échelles. Plus l'échelle augmente, plus la fréquence d'échantillonnage de l'axe des déplacements diminue.

L'axe e peut également être discrétisé, mais, suivant une mesure logarithmique. Les échelles sont logarithmiques car elles demandent une plus grande précision à

basse échelle (c'est-à-dire haute résolution) qu'à haute échelle (c'est-à-dire basse résolution). Le pas doit être fin pour les basses échelles et large pour les hautes échelles. La base du logarithme peut être quelconque.

Cependant, pour une reconstruction correcte, le critère de Nyquist-Shannon (voir paragraphe 3.1.3 page 74) doit être respécté. Ainsi, la base usuellement choisie est la base 2 . Les échelles vont alors prendre les valeurs $1, 2, 4, 8, 16, 32$, etc. Ainsi, lors du passage d'une échelle à la suivante, la fréquence d'échantillonnage de l'axe d peut diminuer de moitié, sans compromettre le critère de Nyquist-Shannon :

$$(2e_1)F_2 = e_1F_1$$

Autrement dit, le pas des déplacements double d'une échelle à la suivante :

$$(2e_1)D_1 = e_1D_2$$

La figure 2.22 illustre cette relation.

Par analogie avec le plan temps-fréquence des séries de Fourier, on parle de *séries d'ondelettes*. La discrétisation des paramètres e et d ne signifie pas que le temps ne reste pas continu... Comme dans les séries de Fourier les indices des coefficients sont entiers et le temps est continu !

Les pas d'échelle et de déplacement s'écrivent en fonction des paramètres *entiers*, j et k :

$$e = e_0^j \text{ et } d = ke_0^jd_0$$

avec $e_0 > 1$ et $d_0 > 0$.

Les ondelettes se réécrivent en fonction de ces paramètres :

$$\psi_{j,k}(t) = e_0^{-j/2}\psi(te_0^{-j} - kd_0)$$

Habituellement, $e_0 = 2$ et $d_0 = 1$:

$$\psi_{j,k}(t) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{t}{2^j} - k\right) \tag{2.11}$$

REMARQUE 2.13.– *Le choix de la base 2 introduit un certain nombre de concepts et de termes que l'on retrouve aussi bien en physique, en musique, que dans l'étude des systèmes auditif et visuel humain. Le fait de doubler l'échelle, pour obtenir une résolution plus basse, revient à diminuer de moitié la fréquence de la basse résolution. On passe alors d'une octave à la suivante et les échelles sont dites dyadiques.*

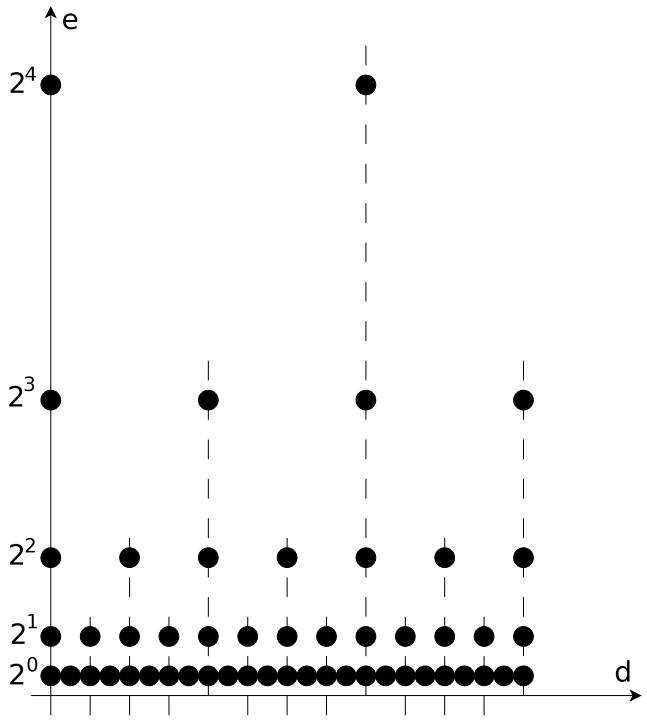


Figure 2.22. *Grille d'échantillonnage de l'espace déplacement-échelle. Les échelles suivent un pas logarithmique en base 2. Les déplacements respectent la contrainte définie par l'équation (2.10)*

2.9. L'analyse multirésolution

Les séries d'ondelettes discrétisent les déplacements (paramètre d) et les échelles (paramètre e). Reste à estimer les intervalles de valeurs des déplacements et des échelles.

On sait que le signal est d'énergie finie :

$$0 < \int_{-\infty}^{+\infty} |s(t)|^2 \mathrm{d}t < +\infty \tag{2.12}$$

Ceci signifie que le spectre $\hat{S}(f)$ est de type passe-bande. Ce spectre est donc borné supérieurement par une fréquence maximale, $f_{\max}$, au-delà de laquelle le signal est d'amplitude nulle. Ainsi, les déplacements sont compris entre 0 et $f_{\max}$.

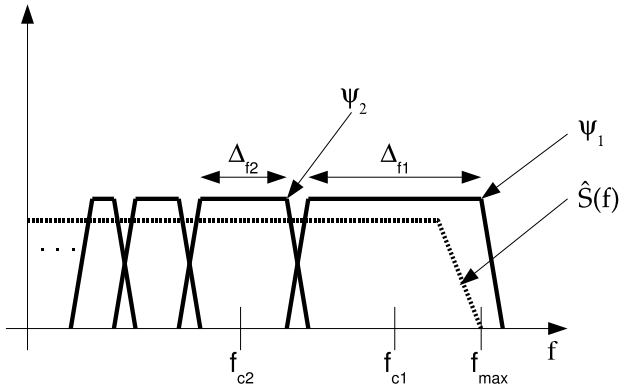


Figure 2.23. *Le recouvrement par les ondelettes peut s'identifier à une analyse par un banc de filtres à Q-constant*

Par ailleurs, les bornes supérieure et inférieure des échelles sont identifiées par l'étude des spectres du signal et des ondelettes. Suivant le principe de Fourier (voir équation 2.9 page 50), une dilatation du temps d'un facteur 2, lors d'un passage d'une échelle à la suivante, équivaut à une compression et à un décalage des fréquences d'un facteur $\frac{1}{2}$. On peut donc couvrir le spectre avec les ondelettes comme l'illustre la figure 2.23 :

1) initialement, on choisit une ondelette, ψ_1, dont le spectre est passe-bande qui couvrent les hautes fréquences, y compris $f_{\max}$. La fréquence centrale de ce spectre est notée f_{c_1} sur la figure ;

2) puis, le facteur d'échelle étant de 2, l'ondelette suivante, ψ_2, est centrée en :

$$f_{c_2} = \frac{1}{2} f_{c_1}$$

et, est de largeur :

$$\Delta f_2 = \frac{1}{2} \Delta f_1$$

3) le procédé est répétée à l'*infini* pour recouvrir l'ensemble du spectre $\hat{S}(f)$ du signal.

On a alors une série d'ondelettes comparables à des *filtres passe-bande* mis à part qu'elles sont en nombre infini.

REMARQUE 2.14.– *Il faut prendre soin que la couverture soit correcte. Les ondelettes doivent légèrement se chevaucher entre elles pour ne pas laisser de trou dans la représentation spectrale.*

REMARQUE 2.15.– *Il faut que le rapport, Q, entre la largeur spectrale, Δf, et la fréquence centrale, f_c, des ondelettes soit constant :*

$$Q = \frac{\Delta f}{f_c} = \text{Cste}$$

En pratique, cette famille ne peut bien sûr pas être infinie. Il faut fixer un niveau maximum, J, de recouvrement par la famille d'ondelettes et laisser l'ensemble des ondelettes au-delà de ce niveau être représenté par une *fonction d'échelle*, φ. Les coefficients de cette fonction, sont de la forme [MAL 89] :

$$\lambda_J(t) = \sum_{j \geqslant J+1} \sum_{k \in \mathbb{Z}} \gamma^{(\varphi)}(j,k) \psi_{j,k}(t)$$

La figure 2.24 schématise cette fonction d'échelle.

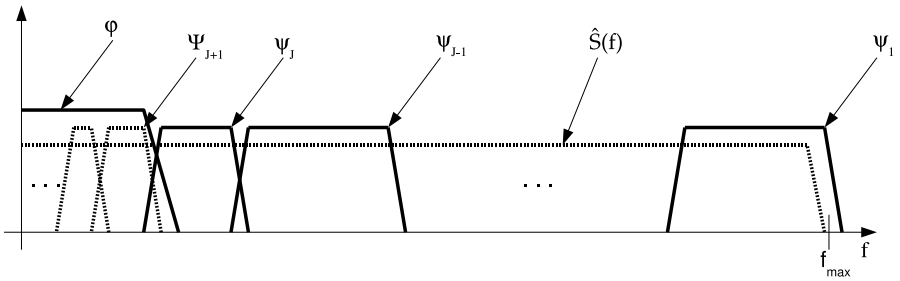


Figure 2.24. *La fonction d'échelle φ recouvre les ondelettes d'indices d'échelle $j > J$*

Les fonctions d'échelle se comportent à l'identique des fonctions d'ondelette. Elles peuvent être déplacées et dilatées.

Stéphane Mallat [MAL 98] a alors introduit une famille, $\{V_i\}$, de sous-espaces de fonctions d'échelle inclus les uns dans les autres :

$$V_i \subset V_{i-1}$$

Le sous-espace V_i est défini par la fonction d'échelle φ_i et ses translatées. L'espace W_i, complémentaire à V_i dans V_{i-1}, est défini par la fonction d'ondelette ψ_i et ses translatées :

$$V_{i-1} = V_i \oplus W_i$$

Il constate alors que la fonction d'échelle φ_i (resp. d'ondelette ψ_i) du sous-espace V_i (resp. W_i) est une combinaison linéaire des translatées de la fonction d'échelle φ_{i-1} de l'espace V_{i-1} :

$$\begin{array}{rcl} \varphi_{i,k}(t) & = & \sum_{m\in\mathbb{Z}} h[m-2k]\varphi_{i-1,k}(t)\forall k \in \mathbb{Z} \\ \psi_{i,k}(t) & = & \sum_{m\in\mathbb{Z}} g[m-2k\varphi_{i-1,k}(t)\forall k \in \mathbb{Z} \end{array}$$

où les coefficients de la combinaison linéaire sont deux filtres :

– $h[m]$ un filtre passe-bas ;

– $g[m]$ un filtre passe-haut.

Ces filtres sont de taille, L, finie : en dehors de cette plage de longueur L, les filtres sont d'amplitude nulle.

REMARQUE 2.16.– *L'annexe A.13 founit de plus amples explications sur les relations entre les fonctions d'échelle, les fonctions d'ondelette et les coefficients d'approximation et de détails.*

Il démontre également que les coefficients $\lambda_i[k]$ (resp. $\gamma_i[k]$), des fonctions d'échelle (resp. d'ondelette) s'obtiennent par convolution des coefficients $\lambda_{i-1}[k]$, avec le filtre passe-bas h (resp. le filtre pass-haut g) :

$$\lambda_i[k] = \sum_{l=0}^{L-1} h[l-2k]\lambda_{i-1}[l] \tag{2.13}$$

$$\gamma_i[k] = \sum_{l=0}^{L-1} g[l-2k]\lambda_{i-1}[l] \tag{2.14}$$

avec la propriété liant le filtre passe-bas, h, au filtre passe-haut, g :

$$g(n) = (-1)^n h(L-(n+1)) \text{ où } L \text{ est la taille des filtres.}$$

Dans les deux équations, on reconnaît une convolution de la séquence λ_{i-1}, des coefficients d'échelle avec les filtres h et g. Toutefois, cette convolution ne conserve qu'un échantillon sur deux. Les filtres h et g sont appliqués tous les $2k$ et le résultat de la convolution est rangé à l'indice k dans la séquence λ_i, des coefficients d'échelle. La séquence à l'échelle i est alors moitié moins longue que la séquence à l'échelle $i-1$.

2.9.1. *L'algorithme d'analyse multirésolution*

A l'aide des équations (2.13) et (2.14), l'algorithme d'analyse multirésolution (AMR) opère récursivement jusqu'à un indice maximum J, de *décomposition*. A chaque étape, la fonction d'échelle est découpée en deux parties égales, à l'aide des filtres passe-bas h et passe-haut g. La fonction d'échelle initiale n'est autre que le signal original. L'indice d'échelle j commence à 0 (pour le signal original) et va en croissant jusqu'à la valeur maximum J.

La figure 2.25 schématise le fonctionnement de l'algorithme AMR d'un signal monodimensionnel. L'opération de sous-échantillonnage est graphiquement représentée par le symbole $\boxed{\downarrow 2}$. Il s'agit simplement d'ignorer un échantillon sur deux; par exemple, conserver uniquement les échantillons pairs.

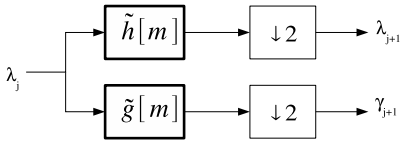


Figure 2.25. *Algorithme MRA : phase d'analyse*

Pour des raisons d'implémentation, la convolution est séparée du sous-échantillonnage :

$$\begin{cases} \tilde{h}[n] &= h[L-(n+1)] \\ \tilde{g}[n] &= g[L-(n+1)] \end{cases} \text{ où } L \text{ est la taille des filtres.}$$

A la résolution la plus basse ($j = J$), la fonction d'échelle fournit une description du signal original privé de ses hautes fréquences, c'est-à-dire de ses détails. C'est donc une approximation du signal à l'échelle J. Les fonctions d'ondelette fournissent les détails perdus. La figure 2.26 illustre le procédé avec un signal de largeur de bande égale à B.

2.9.2. *L'algorithme de synthèse multirésolution*

L'algorithme de synthèse multirésolution devient alors très simple. Il récupère en entrée un signal d'approximation et un signal des détails. A partir de ces deux signaux, il peut reconstruire le signal d'approximation d'échelle inférieur :

$$\lambda_j[k] = \sum_{m=-\infty}^{+\infty} h[k-2m]\lambda_{j+1}[m] + \sum_{m=-\infty}^{+\infty} g[k-2m]\gamma_{j+1}[m] \qquad (2.15)$$

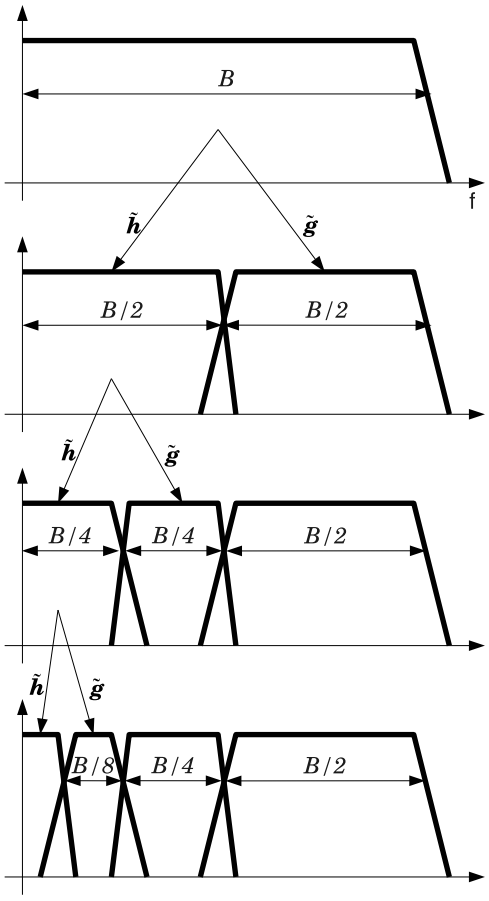


Figure 2.26. *Description du processus récursif de décomposition d'un signal de largeur de bande B à l'aide des filtres passe-bas h et passe-haut g*

Le changement d'échelle implique que le signal final a doublé en taille par rapport aux signaux en données. Ce phénomène est visible dans l'équation (2.15), où les convolutions n'opèrent qu'une fois sur deux $(k - 2m)$ avec les signaux $\lambda_{j+1}[m]$ et $\gamma_{j+1}[m]$.

En termes d'implémentation, il suffit d'insérer des échantillons d'amplitude nulle entre chaque couple d'échantillons des signaux en données, avant d'effectuer les convolutions. La figure 2.27 illustre le procédé. Le symbole $\boxed{\uparrow 2}$ représentent ce prétraitement.

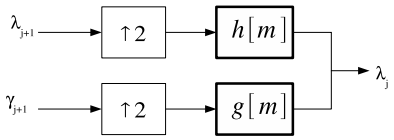


Figure 2.27. *Algorithme MRA : phase de synthèse*

2.9.3. *La multirésolution*

La figure 2.28 montre l'enchaînement d'une analyse à trois niveaux. Les coefficients λ_0, sont les amplitudes du signal original. Les coefficients λ_3, sont ceux du signal d'approximation à l'échelle 3. Les coefficients γ_j, sont ceux des signaux des détails perdus par λ_3.

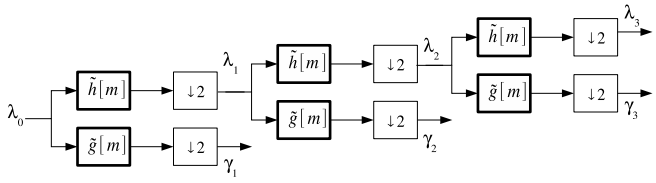


Figure 2.28. *L'analyse à trois niveaux fournit une approximation à l'échelle 3, λ_{j+3}, et des signaux de détails à différentes échelles : γ_{j+3}, γ_{j+2} et γ_{j+1}*

La figure 2.29 montre l'enchaînement d'une synthèse à niveau 3 à partir de l'ensemble des signaux de l'analyse précédente (voir figure 2.28).

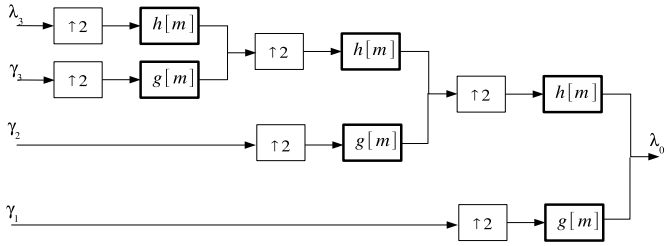


Figure 2.29. *La synthèse à trois niveaux reconstruit le signal λ_0 à partir du résultat de l'analyse à trois niveaux de décomposition : $\{\lambda_3, \gamma_3, \gamma_2, \gamma_1\}$*

2.9.4. *Analyse séparable*

L'AMR appliquée aux images 2D est séparable. Elle s'effectue donc en utilisant les briques de l'algorithme 1D.

La figure 2.30 fournit le schéma algorithmique. L'approximation λ_{j+1}, est obtenue par un filtrage passe-bas, appliqué sur les lignes, puis, sur les colonnes, de l'approximation λ_j.

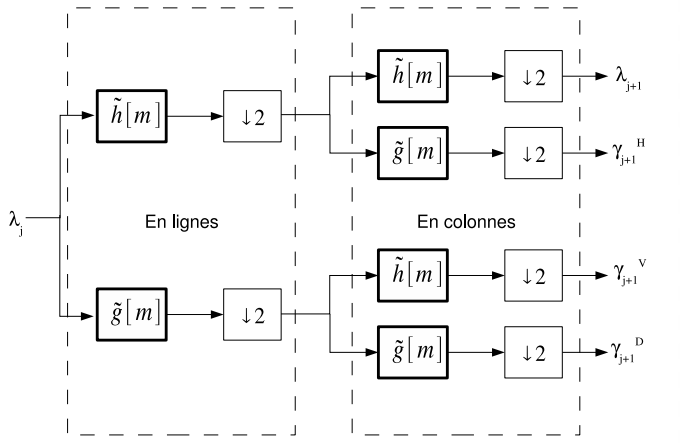


Figure 2.30. *Algorithme MRA des signaux 2D*

Les détails restant sont de trois types :

– un filtrage passe-bas en lignes, suivi d'un filtrage passe-haut en colonnes, de l'approximation λ_j fournit les détails horizontaux γ_{j+1}^H;

– un filtrage passe-haut en lignes, suivi d'un filtrage passe-bas en colonnes, fournit les détails verticaux γ_{j+1}^V;

– un filtrage passe-haut en lignes et en colonnes fournit les détails dits diagonaux γ_{j+1}^D.

Le processus peut être réitéré sur l'approximation λ_{j+1}. Une illustration de la transformée en ondelettes de Haar à trois niveaux est donnée en figure 2.31.

2.10. Un exemple : la transformée de Haar

On va utiliser la transformée de Haar pour illustrer les différents concepts attachés aux ondelettes. Cette transformée est intéressante pour plusieurs raisons :

– son fonctionnement est très intuitif ;

– ses fonctions d'ondelette et d'échelle sont orthonormées. On pourra, ainsi, facilement constater que les mêmes filtres sont utilisés pour l'analyse et pour la synthèse ;

– les graphes des fonctions d'ondelette et d'échelle mettent en évidence l'équation aux deux échelles.

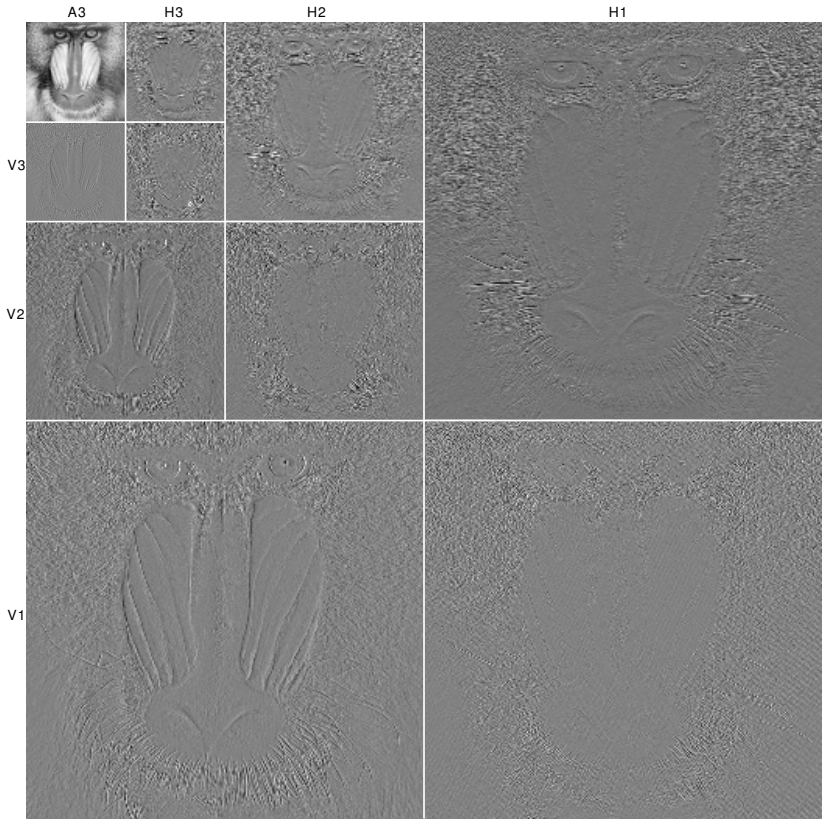


Figure 2.31. *La transformée en ondelettes de Haar du mandrille. L'approximation finale, en haut à gauche, est de niveau 3. Au même niveau de décomposition, dans le sens trigonométrique inverse, et en commençant à droite, on a γ_3^H, γ_3^D et enfin γ_3^V. L'image présentée, qui résulte du collage des dix sous-images de la décomposition sur trois niveaux, est de même taille que l'image originale.*

2.10.1. *Fonctionnement intuitif*

Soit un signal discret temporel $s[t]$ quelconque que l'on veut séparer en une partie basses fréquences et une partie hautes fréquences. Puisque les fréquences basses fournissent une approximation du signal, la moyenne locale définie par :

$$a[t] = \frac{s[2t] + s[2t+1]}{\sqrt{2}}$$

correspond alors aux besoins. Il est à noter que la longueur de l'approximation est diminuée de moitié par rapport à longueur du signal original. Les détails perdus entre

le signal et l'approximation peuvent être retrouvés en utilisant une fonction de différenciation locale :

$$d[t] = \frac{s[2t] - s[2t+1]}{\sqrt{2}}$$

A nouveau, la longueur du signal $d[t]$ est de moitié celle du signal original. Ainsi, cette représentation double $a[t]$ et $d[t]$, est de taille identique au signal $s[t]$.

Il est facile de reconstruire le signal $s[t]$ connaissant $a[t]$ et $d[t]$:

$$\begin{array}{rcl} s[2t] & = & \frac{a[t]+d[t]}{\sqrt{2}} \\ s[2t+1] & = & \frac{a[t]-d[t]}{\sqrt{2}} \end{array}$$

On remarque que les mêmes fonctions sont utilisées pour l'analyse et la synthèse.

2.10.2. *Fonctions d'ondelette et d'échelle*

La fonction d'échelle mère est la fonction caractéristique définie sur $[0, 1]$:

$$\varphi(t) = \begin{cases} 1 & \text{si } 0 \leqslant t \leqslant 1 \\ 0 & \text{sinon} \end{cases}$$

Elle correspond aux fonctions constantes par morceaux. Ses filles se calculent suivant l'équation A.3 de l'annexe A.13.

Le filtre numérique $h_\varphi[n]$ a pour coefficients :

$$h_\varphi[n] = \begin{cases} 1/\sqrt{2} & \text{si } n = 0 \text{ ou } n = 1 \\ 0 & \text{sinon} \end{cases}$$

Et le filtre numérique $g_\varphi[n]$:

$$g_\varphi[n] = \begin{cases} 1/\sqrt{2} & \text{si } n = 0 \\ -1/\sqrt{2} & \text{si } n = 1 \\ 0 & \text{sinon} \end{cases}$$

On n'oublie pas que la séparation en basses et hautes fréquences – par convolution avec les filtres h_φ et g_φ – est suivie d'un sous-échantillonnage (voir figure 2.25).

2.10.3. *Jeu d'essai*

Soit le signal $s[t]$ défini par la séquence de valeurs :

$$s[t] = [125 \quad 12 \quad 45 \quad 89 \quad 435 \quad 4 \quad 78 \quad 96]$$

On veut le décomposer jusqu'au niveau $J = 3$.

2.10.3.1. *L'approche intuitive*

Le signal est l'approximation d'un phénomène réel à la meilleure résolution possible (celle du capteur ayant fait la mesure). Il est communément noté comme le niveau de décomposition le plus faible : $s[t] = a_0[t]$. A partir de ce signal, on calcule les signaux d'approximation et de détail du niveau 1 :

$$\begin{array}{rcl} a_1[t] & = & [96,8 \quad 94,7 \quad 310,4 \quad 123] \\ d_1[t] & = & [79,9 \quad -31,1 \quad 304,7 \quad -12,7] \end{array}$$

Puis, on décompose l'approximation $a_1[t]$:

$$\begin{array}{rcl} a_2[t] & = & [135,5 \quad 306,5] \\ d_2[t] & = & [1,5 \quad 132,5] \end{array}$$

Enfin, on procède de même avec $a_2[t]$:

$$\begin{array}{rcl} a_3[t] & = & [312,5] \\ d_3[t] & = & [-120,9] \end{array}$$

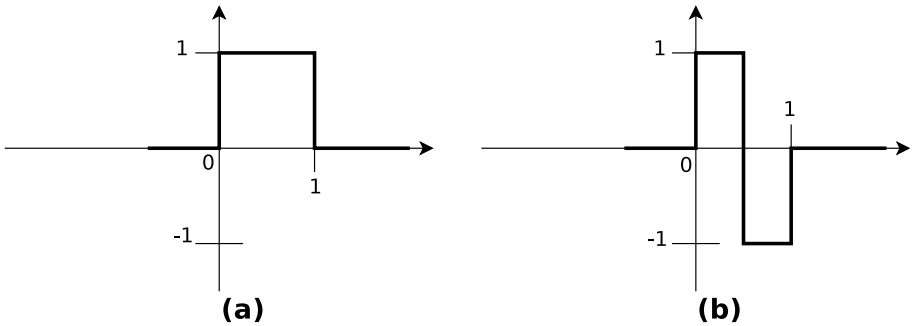


Figure 2.32. *(a) Fonctions d'échelle et (b) d'ondelette de la transformée de Haar au coefficient $\frac{1}{\sqrt{2}}$ près*

Les graphes des $a_i[t](i = 0, 1, 2)$ sont fournis en figure 2.33.

Le signal $s[t]$ peut alors être représenté par le signal d'approximation de plus faible résolution, $a_3[t]$, et l'ensemble des signaux de détails, $d_3[t]$, $d_2[t]$ et $d_1[t]$:

$$Ws[t] = [312,5 \quad -120,9 \quad 1,5 \quad 132,5 \quad 79,9, -31,1 \quad 304,7 \quad -12,7]$$

Cette représentation est de taille identique à celle du signal original. Elle permet de reconstruire à l'identique le signal original ainsi que toutes les approximations intermédaires.

2.10.3.2. *L'approche MRA*

On va maintenant suivre pas à pas l'algorithme MRA pour constater que le résultat, *in fine*, est identique.

1) le signal $a_0[t]$, est convolué avec les filtres h_φ et g_φ. On obtient les deux signaux :

$$\begin{array}{rcl} a_1[t] & = & [96,8 \quad 40,3 \quad 94,7 \quad 370,5 \quad 310,4 \quad 57,9 \quad 123] \\ d_1[t] & = & [79,9 \quad -23,3 \quad -31,1 \quad -244,6 \quad 304,7 \quad -52,3 \quad -12,7] \end{array}$$

2) ensuite ces deux signaux sont sous-échantillonnés :

$$\begin{array}{rcl} a_1[t] & = & [96,8 \quad 94,7 \quad 310,4 \quad 123] \\ d_1[t] & = & [79,9 \quad -31,1 \quad 304,7 \quad -12,7] \end{array}$$

3) puis les étapes (1) et (2) sont appliquées sur le signal $a_1[t]$:

$$\begin{array}{rcl} a_2[t] & = & [135,5 \quad 306,5] \\ d_2[t] & = & [1,5 \quad 132,5] \end{array}$$

4) enfin, ces étapes sont appliquées sur le signal $a_2[t]$:

$$\begin{array}{rcl} a_3[t] & = & [312,5] \\ d_3[t] & = & [-120,9] \end{array}$$

On retrouve la même décomposition $\mathrm{Ws}[t]$, que par l'approche intuitive.

2.11. Les ondelettes de seconde génération

La théorie des transformées en ondelettes et des bancs de filtres est loin de se limiter aux aspects que l'on vient, succintement, de présenter. A partir du simple exemple

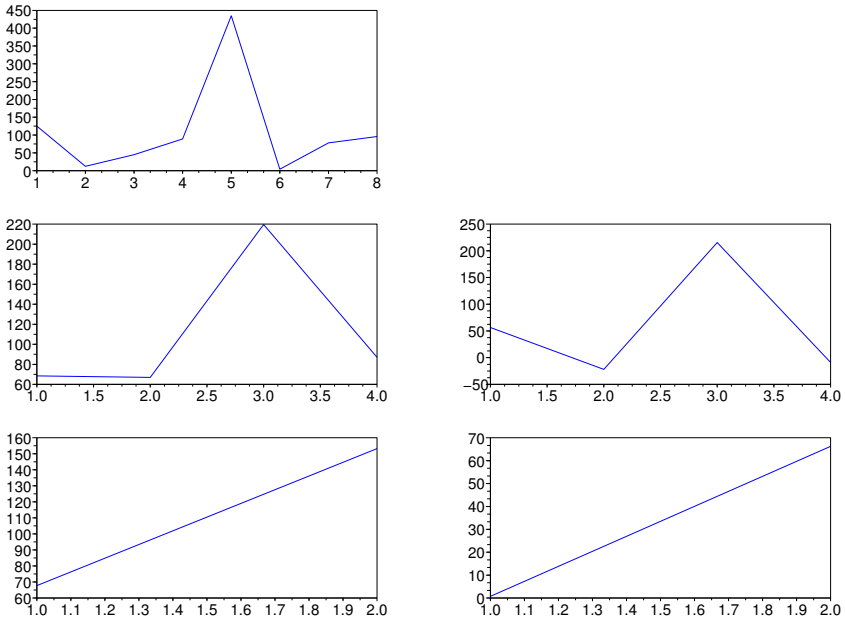


Figure 2.33. *Décomposition du signal $s[t] = a_0[t]$ présenté en première ligne : chaque ligne est un niveau de décomposition appliquée sur le signal d'approximation de la ligne supérieure; le signal d'approximation est à gauche et celui des détails à droite*

précédent, il est aisé de constater que l'approche intuitive est plus efficace que l'approche MRA. Les fonctions utilisées par l'approche intuitive effectuent *simultanément* la convolution et le sous-échantillonnage.

Partant de cette constatation, une *seconde génération* d'ondelettes est alors apparue depuis quelques années. Non seulement cette nouvelle génération est algorithmiquement plus efficace, mais, elle généralise l'approche précédente. Les deux principaux intérêts sont :

– plus de souplesse dans la définition des ondelettes à utiliser ;

– des algorithmes de décomposition et de reconstruction plus efficaces.

On ne s'attardera pas sur les définitions liées à cette seconde génération. Le lecteur intéressé pourra consulter le site de Clemens Valens qui offre une introduction remarquable ainsi que les références nécessaires : http://pagesperso-orange.fr/polyvalens/clemens/lifting/lifting.html. On va simplement montrer comment cette nouvelle approche effectue la transformée de Haar.

Partant de l'analyse de l'approche intuitive, on sépare notre signal x, en deux parties : les échantillons pairs x_e, d'un côté et les échantillons impairs x_o, de l'autre. Ensuite, sachant que le signal x, est corrélé, on émet une prédiction P [7] :

$$d = x_o - P(x_e)$$

Si l'on prend comme prédiction l'identité $P(x) = x$, d enregistre alors la différence entre les échantillons pairs et les échantillons impairs.

On peut ainsi décrire le signal x, par x_o et d; on obtient les échantillons pairs par simple soustraction :

$$x = x_o + x_e = x_o + (x_o - d) = 2x_o - d$$

Dans ce cas, le signal x est approximativement décrit par ses échantillons impairs. Cette approximation x_o, risque de ne pas être très fidèle au signal x. Pour l'améliorer, on effectue une *mise-à-jour* U, des échantillons pairs :

$$s = x_e + U(d)$$

Si l'on choisit comme mise-à-jour $U(x) = x/2$, on retrouve la définition de la transformée de Haar en termes de *schéma lifting* où s est l'approximation (moyenne des échantillons pairs et impairs) et d la différence entre s et x.

L'algorithme, dans sa version générique, est présenté en figure 2.34.

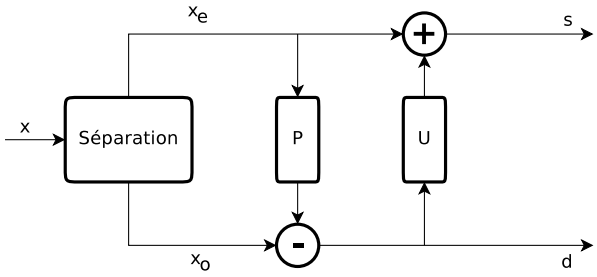


Figure 2.34. *Analyse par schéma lifting*

La synthèse suit le même principe algorithmique en sens inverse comme le montre la figure 2.35.

7. C'est un procédé très efficace que l'on développera avec les DPCM du prochain chapitre.

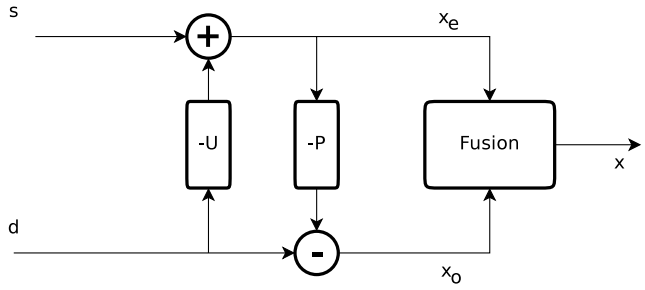


Figure 2.35. *Synthèse par schéma lifting*

Cette base de schéma *lifting*, appelée *lifting* par *ondelette fainéante*, sert de support à la construction de schémas de *lifting* plus élaborés – mais pas plus complexes dans leurs algorithmes – permettant d'incorporer toute sorte d'ondelettes. Ce sujet sort du cadre de ce livre, on n'en dira pas plus.

REMARQUE 2.17.– *Lors de l'utilisation des ondelettes pour la compression d'images et de vidéos, les algorithmes sous-jacents seront de type schéma* lifting.

2.12. Synthèse

Ce chapitre vient décrire les transformées liées au domaine fréquentiel. On a vu une description générale des techniques utilisées et on a montré l'intérêt des approches combinant une observation dans l'espace original à une observation fréquentielle. Cette combinaison permet de localiser des fréquences et, ainsi, de mieux estimer si l'on doit les conserver ou les supprimer.

Ce chapitre est une introduction aux théories des transformées que des citations bibliographiques, régulièrement fournies, viennent compléter.

Tout au long de ce chapitre, on a continuellement fait référence aux signaux dans leurs versions numériques (ou discrètes) sans expliquer le principe de la numérisation ni les propriétés qui en découlent. Aussi, le prochain chapitre aborde les thèmes de la numérisation[8], de la quantification et du codage qui définissent et caractérisent les signaux discrets.

8. *digitalisation* par anglicisme.

Chapitre 3

Numérisation, quantification et codage

Ce deuxième chapitre présente, en section 3.1, la conversion des signaux analogiques en signaux numériques (CAN), ainsi que la conversion inverse – du numérique vers l'analogique – (CNA). Les deux processus de ces conversions sont l'*échantillonnage* et la *quantification*.

Ensuite, la section 3.2 introduit succintement la *théorie de l'information*, en insistant sur les conséquences pour le *codage* (voir paragraphe 3.2.1.2) et la *quantification* (voir paragraphe 3.2.1.3).

Puis, la section 3.3 décrit les quantifications habituellement utilisées. Le principe étant de remplacer l'alphabet de la source par un alphabet réduit en taille, une quantification entraîne des pertes plus ou moins importantes.

Les *codeurs entropiques*, présentés en section 3.4, cherchent également à compresser l'information, mais, en évitant toute perte.

Avant de conclure, la section 3.5 présente les techniques de prédiction qui s'avèrent être aussi efficaces qu'elles sont simples à implémenter.

3.1. Numérisation

La numérisation ou, par anglicisme, la digitalisation (de sigle PCM[1]), des signaux analogiques constitue une étape importante dans le transport de l'information. En

1. *Pulse Code Modulation* (MIC : modulation d'impulsions codées).

effet, un signal numérisé, qu'il soit audio, vidéo ou autre, se présente en une séquence de bits pouvant être manipulée à souhait sans aucune perte. Mais, il faut s'assurer que les valeurs codées par la séquence restent fidèles aux valeurs continues du signal analogique pouvant exprimer, par exemple, des tensions électriques (mesurées en volts).

Les appareils de conversion des signaux analogiques en signaux numériques (CAN[2]), opèrent en deux étapes : l'*échantillonnage* et la *quantification*.

L'*échantillonnage* consiste à sélectionner les valeurs qui vont représenter le signal. Le paragraphe 3.1.3 en donne les principes et les propriétés.

Les échantillons ainsi récupérés ont des valeurs dans le domaine continu qu'il convient de traduire en valeurs discrètes; c'est-à-dire représentables par une séquence de bits. Cette discrétisation des amplitudes est appelée *quantification*. Elle ne sera étudiée qu'en section 3.3 car elle est utilisée à d'autres étapes de la compression. En effet, les canaux de transmission et les supports de stockage ayant des capacités limitées, la quantification est également utilisée pour minimiser le volume d'occupation de l'information afin de ne pas saturer les supports de stockage ni d'encombrer les réseaux.

Une fois les étapes d'échantillonnage et de quantification effectuées, on obtient un signal numérique. Si les théorèmes d'échantillonnage et de quantification sont respectées, alors le signal analogique original peut être reconstruit à l'identique. Cette reconstruction est l'objet des appareils de conversion inverse des signaux numériques en signaux analogiques (CNA[3]).

Mais, avant d'étudier l'échantillonnage, le paragraphe 3.1.1 présente la classification usuelle des types de signaux. Puis le paragraphe 3.1.2 détaille les deux signaux qui ont une place prépondérante dans le processus d'échantillonnage : le *signal porte* et l'*impulsion de Dirac*.

3.1.1. *Définitions et propriétés*

Afin de connaître les propriétés des signaux, ceux-ci sont répartis en catégories répondant à différentes situations. La figure 3.1 présente cette classification. En première classification, les signaux sont déterministes ou aléatoires. Les *signaux déterministes* sont prévisibles dans le temps ou, plus généralement, dans l'espace[4]. Ils

2. Convertisseur Analogique/Numérique.

3. Convertisseur numérique/analogique.

4. Ce chapitre ne présente que les signaux monodimensionnels, temporels. Mais l'ensemble des définitions données dans ce chapitre se généralise aisément aux dimensions supérieures et

peuvent donc être modélisés mathématiquement. En revanche, les *signaux aléatoires* sont imprévisibles et seuls des observations statistiques permettent de les appréhender.

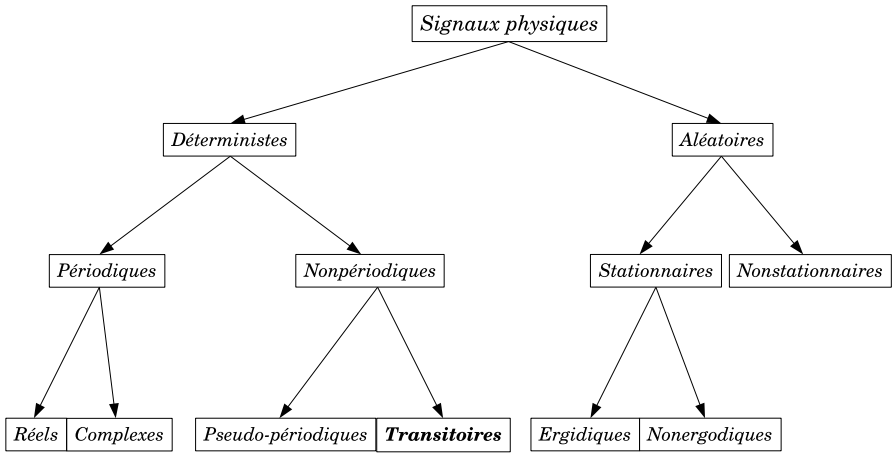


Figure 3.1. *Classification temporelle des signaux*

Les signaux déterministes peuvent être *périodiques* ou pas. Les signaux périodiques sont *complexes* ou *réels*. Dans le chapitre traitant des transformées (voir chapitre 2), on a déjà utilisé les signaux périodiques. Lorsque les signaux sont non-périodiques, ils peuvent être soit *pseudo-périodiques*, soit *transitoires*. Les signaux pseudo-périodiques sont décrits par des sommes de signaux périodiques de différentes périodes, alors que les signaux *transitoires* ont une existence limitée dans le temps. En théorie, l'ensemble de ces signaux déterministes sont reproduisibles à l'identique autant de fois que voulu.

La deuxième grande catégorie, les signaux *aléatoires*, se divise en deux sous-catégories : les signaux *stationnaires* et les signaux *nonstationnaires*. Les signaux stationnaires ont pour propriété principale d'avoir leurs valeurs moyennes indépendantes du temps. Un signal est dit *ergodique* lorsque la moyenne statistique, estimée à un instant donné sur plusieurs réalisations du signal, est égale à la moyennne calculée à partir d'un seul de ces essais, mais, durant une période de temps adéquate.

Les signaux physiques utilisés par le multimédia – principalement, l'audio, l'image et la vidéo – sont rangés dans la catégorie des signaux transitoires.

---

en particulier au cas bidimensionel (par exemple les images) ou au cas tridimensionnel (par exemple la vidéo qui est une séquence temporelle d'images 2D).

3.1.1.1. *Description énergétique*

La classification des signaux peut également être énergétique. Dans ce cas, on peut distinguer les signaux d'énergie finie :

$$\int_{-\infty}^{+\infty} s^2(t)\mathrm{d}t < +\infty$$

des signaux de puissance moyenne finie :

$$0 \leqslant \lim_{T\to+\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} s^2(t)\mathrm{d}t < +\infty$$

L'interprétation des termes d'énergie et de puissance est donnée en annexe B.1.

Il est à noter que :

– les signaux de puissance moyenne non nulle sont d'énergie infinie ;

– les signaux d'énergie finie sont de puissance moyenne nulle.

Les signaux physiques sont de la deuxième catégorie. Aussi, par la suite, on parlera de signaux d'énergie finie ou de puissance moyenne nulle.

3.1.1.2. *Description spectrale*

Les signaux peuvent également être classés suivant leurs valeurs spectrales. Dans le domaine fréquentiel, la distribution de l'énergie ou de la puissance occupe une plage de fréquences, appelée *bande*, définie par sa largeur (voir figure 3.2) :

$$\Delta F = F_{\max} - F_{\min}$$

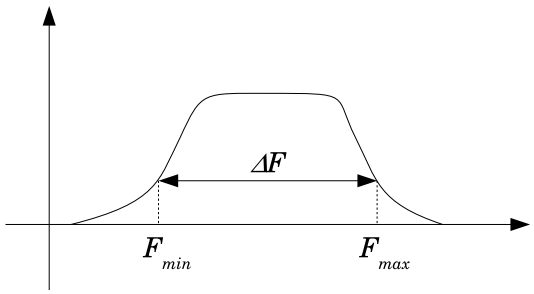


Figure 3.2. *Largeur de bande*

Un signal peut être caractérisé à bande étroite ou à large bande; à partir de sa largeur de bande et de sa fréquence moyenne :

$$F_{\mathrm{moy}} = \frac{(F_{\max} - F_{\min})}{2}$$

– si le rapport $\frac{\Delta F}{F_{\text{moy}}}$ est faible, le signal est à *bande étroite* ;

– si le rapport $\frac{\Delta F}{F_{\text{moy}}}$ est élevé, il est à *large bande*.

Les signaux à bandes étroites sont identifés par leurs fréquences moyennes :

– les signaux basses fréquences (BF) sont de fréquence moyenne F_{moy} inférieure à $250\,\text{kHz}$;

– les signaux hautes fréquences (HF) sont de $F_{\text{moy}} \in [250\,\text{kHz}, 30\,\text{MHz}]$;

– les signaux très hautes fréquences (VHF) sont de $F_{\text{moy}} \in [30\,\text{MHz}, 300\,\text{MHz}]$;

– les signaux ultra hautes fréquences (UHF) sont de $F_{\text{moy}} \in [300\,\text{MHz}, 3\,\text{GHz}]$;

– les signaux super hautes fréquences (SHF) sont de $F_{\text{moy}} > 3\,\text{GHz}$.

Parmi les signaux à larges bandes, on trouve :

– les signaux lumineux ultraviolets dont la longueur d'onde $\lambda_{\text{moy}} = \frac{1}{F_{\text{moy}}} \in [10\,\text{nm}, 400\,\text{nm}]$;

– les signaux lumineux visibles dont $\lambda_{\text{moy}} \in [400\,\text{nm}, 700\,\text{nm}]$;

– les signaux lumineux infrarouges dont $\lambda_{\text{moy}} \in [700\,\text{nm}, 1000\,\text{nm}]$.

On retrouvera les signaux lumineux visibles (pour l'humain) dans le chapitre 4 qui traite des espaces couleurs.

3.1.1.3. *Principe de la numérisation*

Les techniques d'estimation et de suppression des *bruits*, de par nature aléatoires, venant perturber l'information originale, sont ignorées. Les signaux étudiés sont considérés déterministes.

Lors de la numérisation, un signal analogique (à temps continu) est représenté par un certain nombre d'*échantillons*, espacés dans le temps. L'extraction de ces valeurs discrètes est appelée *échantillonnage* du signal analogique. Les amplitudes étant également continues, il faut également les décrire par des valeurs discrètes. Cette *quantification* divise la plage des amplitudes en un ensemble d'intervalles. Chaque intervalle est représenté par une valeur qui lui est significative, appelée *représentant*. Un échantillon prend alors pour amplitude discrète le représentant de l'intervalle auquel il appartient.

La numérisation est principalement composée de deux processus qui peuvent être appliqués dans n'importe quel ordre. La figure 3.3 schématise les résultats intermédiaires des étapes de la numérisation.

Pour que la numérisation soit correcte, il faut s'assurer que la *reconstruction par interpolation* du signal numérique fournit le signal analogique initial. Mais avant d'aller plus en détail avec ce principe fondamental, deux signaux essentiels sont présentés.

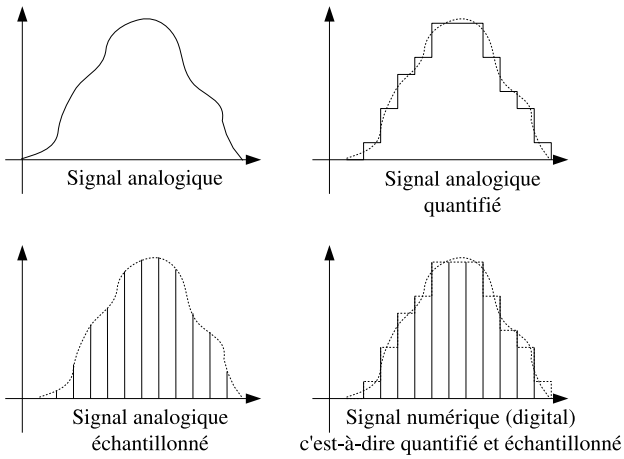


Figure 3.3. *Echantillonnage et quantification : les deux processus de la numérisation*

3.1.2. *Deux signaux particuliers*

Certains signaux ont des propriétés qui leur confèrent une place prépondérante dans le cadre du traitement du signal numérique :

– le *signal rectangulaire*, ou fonction porte, $\Pi_\tau(t)$;

– l'*impulsion de Dirac* – aussi appelée distribution ou fonction généralisée de Dirac.

3.1.2.1. *Signal restangulaire*

Le signal rectangulaire, ou fonction porte, $\Pi_\tau(t)$ est défini par l'équation suivante (voir figure 3.4) :

$$\Pi_\tau(t) = \begin{cases} A & \text{si} |t| \leqslant \tau \\ 0 & \text{sinon} \end{cases}$$

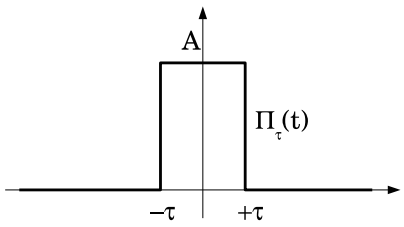


Figure 3.4. *Le signal rectangulaire* $\Pi_\tau(t)$

L'intérêt d'un tel signal est de découper un signal analogique en tronçons plus ou moins fins. Ainsi, les tronçons peuvent être traités indépendamment les uns des autres. La transformée de Fourier $\hat{\Pi}_\tau(f)$, de ce signal est un sinus cardinal (voir annexe B.2) :

$$\hat{\Pi}_\tau(f) = 2A\tau \,\mathrm{sinc}(2f\tau)$$

Comme le montre la figure 3.5, cette fonction n'opère pas une coupure franche dans le domaine fréquentiel, mais, effectue un lent amortissement des hautes fréquences.

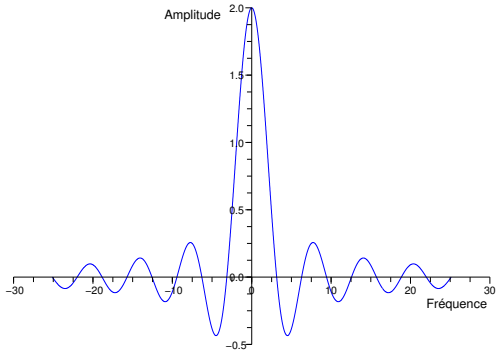


Figure 3.5. *La transformée de Fourier de* $\Pi_\tau(t)$ *avec* $\tau = \frac{1}{2}$ *et* $A = 2$

3.1.2.2. *Impulsion de Dirac*

L'impulsion de Dirac n'est pas une fonction classique, mais, une *fonction généralisée* [GAS 00]. Elle définit la base du traitement du signal numérique.

L'impulsion de Dirac peut se formuler comme le passage à la limite de la fonction rectangle, $\Pi_\tau(t)$ avec $A = \frac{1}{2\tau}$, quand τ tend vers 0 :

$$\delta(t) = \lim_{\tau \to 0} \left(\Pi_\tau(t)|_{A=\frac{1}{2\tau}} \right)$$

Cette impulsion et sa représentation fréquentielle, $\hat{\delta}(f) = \mathbb{1}$ (voir annexe B.3), sont schématisées comme le montre la figure 3.6.

Il existe d'autres définitions de cette impulsion :

$$\begin{cases} \delta(t) = 0 & \forall t \neq 0 \\ \int_{-\infty}^{+\infty} \delta(t)\mathrm{d}t = 1 \end{cases}$$

Ou encore :

$$\begin{cases} \int_{-\infty}^{+\infty} x(t)\delta(t)\mathrm{d}t = x(0) \\ \text{où } x \text{ est une fonction quelconque sans discontinuité en } 0. \end{cases} \tag{3.1}$$

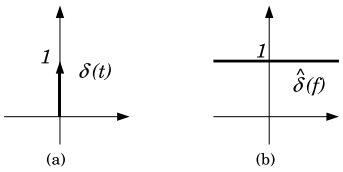


Figure 3.6. *(a) L'impulsion de Dirac $\delta(t)$, et (b) sa représentation fréquentielle $\hat{\delta}(f)$*

3.1.3. *Echantillonnage*

Avec la définition de l'équation (3.1), apparaît tout l'intérêt de l'impulsion de Dirac. Si l'impulsion est translatée de t_0, $\delta(t - t_0)$, cette même définition donne la valeur de $x(t_0)$. Ainsi, en regroupant plusieurs impulsions de Dirac, *régulièrement* espacées, d'un pas T_e, on obtient un *peigne de Dirac*. La fonction $x(t)$ multipliée par ce peigne, fournit les valeurs de $x(t)$ régulièrement espacées, d'un pas T_e. La figure 3.7a donne le schéma usuel d'un peigne de Dirac et la figure 3.7b, l'effet du produit de ce peigne avec une fonction continue $x(t)$. C'est le *schéma théorique d'échantillonnage*.

L'analogie peut être faite entre un peigne de Dirac et un scanner d'ordinateur, constitué d'une barrette de pastilles de silicium photosensibles régulièrement espacées. Chaque élément de silicium peut être modélisé par une impulsion de Dirac. Une barrette de silicium est donc modélisée par un peigne de Dirac.

Le peigne de Dirac s'écrit :

$$\Delta_{\mathrm{Te}}(t) = \sum_{n\in\mathbb{Z}} \delta(t - nT_e)$$

Sa tranformée de Fourier est également un peigne de Dirac d'amplitude et de période $F_e = 1/T_e$ (cf. annexe B.4) :

$$\hat{\Delta}_{T_e}(f) = F_e \sum_{n\in\mathbb{Z}} \delta(f - \mathrm{nF}_e)$$

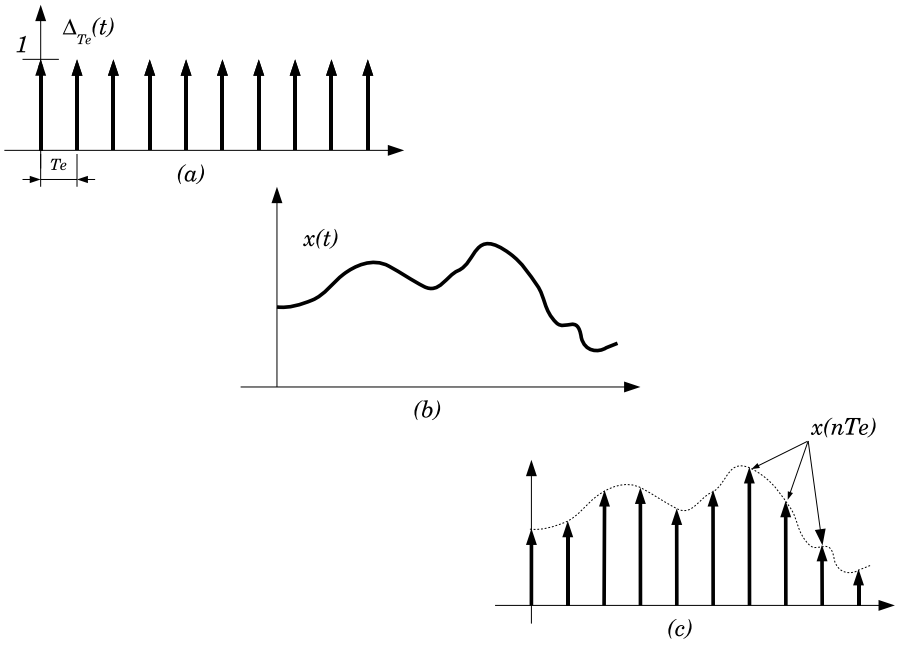


Figure 3.7. *(a) Le peigne de Dirac, $\Delta_{T_e}(t)$; (b) le signal analogique $x(t)$; (c) les échantillons, $x(nT_e)$*

THÉORÈME 3.1.– *Un signal continu $x(t)$ multiplié par un peigne de Dirac founit une séquence, $y(t)$, de valeurs appelées échantillons :*

$$\begin{aligned} y(t) &= x(t)\Delta_{T_e}(t) \\ &= x(t)\sum_{n\in\mathbb{Z}}\delta(t-nT_e) \\ &= \sum_{n\in\mathbb{Z}}x(t)\delta(t-nT_e) \\ &= \sum_{n\in\mathbb{Z}}x(nT_e) \end{aligned}$$

D'après le théorème de convolution (voir théorème 2.1 page 30), la transformée de Fourier $\hat{Y}(f)$, de la séquence d'échantillons est égale au *produit de convolution* de la transformée de Fourier $\hat{X}(f)$, du signal source avec la transformée de Fourier $\hat{\delta}(f)$, du peigne :

$$\hat{Y}(f) = \hat{X}(f) \otimes \left(F_e \sum_{n\in\mathbb{Z}} \delta(f - nF_e) \right)$$

Comme l'impulsion de Dirac est l'élément neutre de la convolution :

$$\hat{Y}(f) = F_e \sum_{n\in\mathbb{Z}} \hat{X}(f - nF_e)$$

Par conséquent, le spectre $\hat{Y}(f)$, du signal échantillonné, duplique le spectre $\hat{X}(f)$, du signal analogique, avec un pas de valeur F_e. La figure 3.8 illustre le phénomène.

Il faut prendre garde à la fréquence d'échantillonnage F_e, utilisée. Si le même signal analogique que celui de la figure 3.8 est échantillonné, mais, avec une fréquence inférieure à celle de la fréquence maximum du signal analogique, il apparaît un *repliement spectral*.

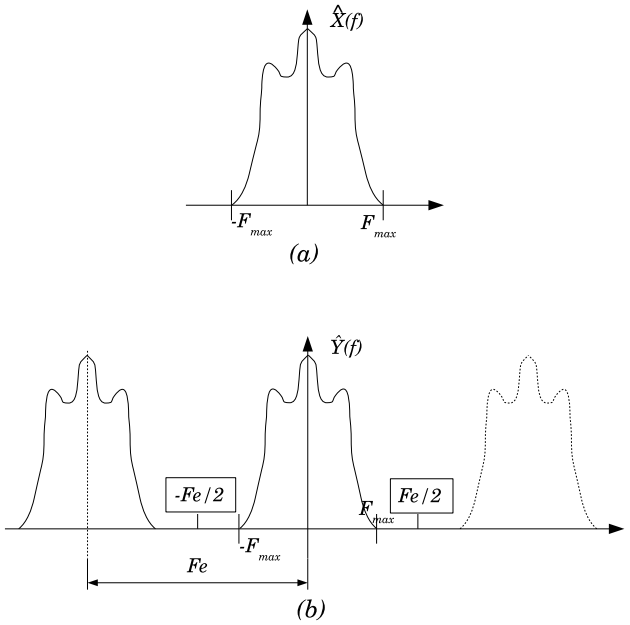


Figure 3.8. *(a) le spectre $\hat{X}(f)$, du signal analogique. (b) le spectre $\hat{Y}(f)$, du signal échantillonné*

Ce phénomène, illustré en figure 3.9, est un *recouvrement* des hautes fréquences par les basses fréquences. Le spectre du signal échantillonné ne permet plus d'identifier le spectre du signal analogique.

THÉORÈME 3.2.– *Harry Nyquist et Claude Shannon ont montré que pour assurer une reconstruction exacte d'un signal analogique à partir de sa version échantillonnée, l'échantillonnage doit avoir été fait à une fréquence suffisamment élevée pour éviter tout repliement spectral :*

$$F_e \geqslant 2f_{\max}$$

avec F_e la fréquence d'échantillonnage et $f_{\max}$ la fréquence maximum du signal analogique.

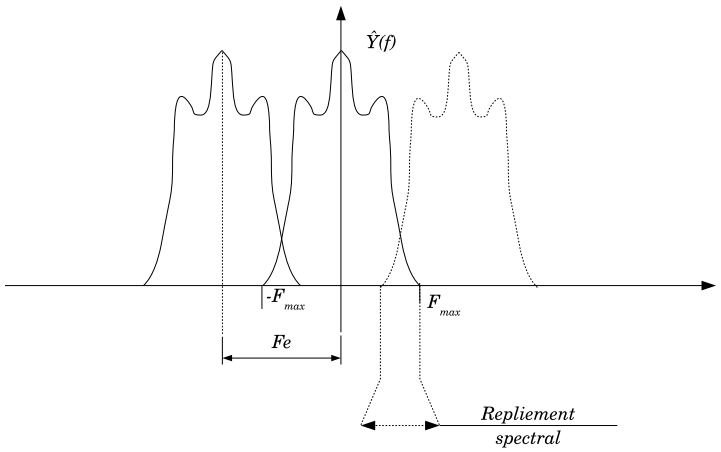


Figure 3.9. *Lorsque la fréquence d'échantillonnage F_e, est inférieure à $2f_{\max}$ il apparait un repliement spectral*

REMARQUE 3.1.– *Malheureusement, la fréquence maximale $f_{\max}$, du signal analogique n'est pas toujours connue ou alors celle-ci est trop élevée pour permettre un échantillonnage correct.*

Pour y remédier, il est fait usage d'un filtre passe-bas qui élimine les hautes fréquences au-delà d'une fréquence dite de coupure F_C. Il est supposé que ces hautes fréquences sont peu importantes pour l'analyse et la synthèse du signal. La fréquence d'échantillonnage F_e, est alors fonction de la fréquence de coupure F_C.

Lors de la reconstruction du signal analogique, il faut également éliminer les fréquences supérieures à la fréquence de coupure. Ceci est fait en réutilisant le même filtre passe-bas.

Si le théorème 3.2 est respecté, il suffit de récupérer la partie du spectre échantillonné $\hat{Y}(f)$, comprise dans l'intervalle $[-\frac{F_e}{2}, +\frac{F_e}{2}]$ pour retrouver le spectre $\hat{X}(f)$, du signal analogique. Ainsi, la reconstruction est exacte.

En revanche, si le théorème n'est pas respecté, un repliement spectral se produit qui se traduit par une déformation lors de la reconstruction du signal.

Dans la figure 3.10, le signal original $x(t)$, est un sinus de fréquence $5\,\mathrm{kHz}$ qui est *sous-échantillonné* à $8\,\mathrm{kHz}$ ce qui provoque un repliement spectral puisque la fréquence d'échantillonnage est inférieure à la fréquence de $10\,\mathrm{kHz}$ imposée par le signal. Ce recouvrement se traduit temporellement par un signal apparant de plus basse fréquence (c'est-à-dire de plus longue période).

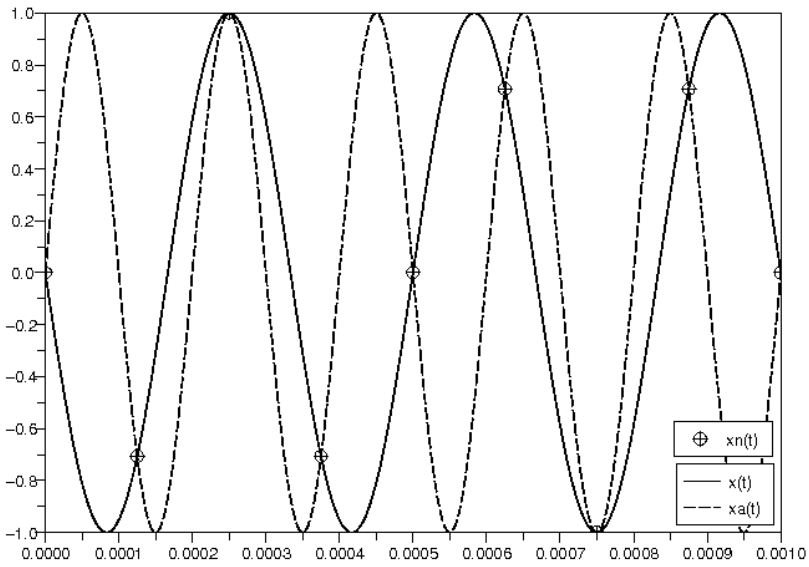


Figure 3.10. *Quand l'échantillonnage ne respecte pas le théorème 3.2, la synthèse du signal n'est pas correcte. Sur le graphe, le signal originel $x(t)$, en trait plein, est un sinus de fréquence $5\,\mathrm{kHz}$. Il est échantillonné à $8\,\mathrm{kHz}$ ($< 2 * 5\,\mathrm{kHz}$). Les échantillons sont indiqués sur le graphe par des $\oplus$. Le signal $x_a(t)$ reconstruit à partir de ces échantillons est en trait pointillé sur le graphe.*

3.1.4. *Synthèse de la numérisation*

L'échantillonnage est une étape importante. Sans lui, le signal numérique n'existerait pas. Cette section n'a donné qu'un aperçu de la théorie et des problèmes techniques qui en découlent. Le lecteur intéressé peut se référer, dans une première lecture, aux deux excellents livres d'introduction sur le sujet de Françis Cottet [COT 02] et de A.W.M. Van Den Enden et N.A.M.Verhœckx [VAN 03].

Mais un signal numérique n'est pas seulement qu'un signal analogique échantillonné. Les amplitudes de ce dernier s'expriment souvent sur une échelle réelle qu'il convient de quantifier.

Habituellement, les capteurs électroniques délivrent des amplitudes analogiques voltaïques qui doivent être transcrites en nombres réels *machines*. Les amplitudes passent donc du domaine continu au domaine discret (car les réels machines ne sont pas des réels mathématiques !). Ce traitement concerne la quantification qui est étudiée en section 3.3.

3.2. Théorie de l'information

Cette section est la base théorique de la quantification et du codage entropique. Elle fait continuellement référence aux notions d'alphabet et de distribution de probabilité associées aux éléments de cet alphabet.

Soit l'alphabet A, qui fournit l'ensemble des valeurs qu'un émetteur produit en direction d'un récepteur. Ces valeurs, appelées *événements*, surviennent plus ou moins fréquemment dans la source émise. Relativement au nombre total d'événements, ces fréquences définissent les *probabilités* des événements. Elles sont d'une importance capitale pour la gestion du coût matériel et du délai de transmission, lors de l'émission de la source.

EXEMPLE 3.1.– *Dans le jeu du scrabble, le nombre de jetons par lettre de l'alphabet n'est pas uniforme. La fréquence des lettres dépend de la langue choisie. Ainsi, La version française diffère de la version anglaise.*

La probabilité d'une lettre est alors le rapport entre le nombre de jetons la représentant et le nombre total de jetons. Par exemple, la lettre E a le plus grand nombre de jetons, alors que la lettre Z n'en possède qu'un. Le tableau ci-après donne les probabilités de toutes les lettres.

Lettre	*Nombre de jetons*	*Probabilité*	*Lettre*	*Nombre de jetons*	*Probabilité*
$'A'$	*9*	*0,09*	$'N'$	*6*	*0,06*
$'B'$	*2*	*0,02*	$'O'$	*6*	*0,06*
$'C'$	*2*	*0,02*	$'P'$	*2*	*0,02*
$'D'$	*3*	*0,03*	$'Q'$	*1*	*0,01*
$'E'$	*15*	*0,15*	$'R'$	*6*	*0,06*
$'F'$	*2*	*0,02*	$'S'$	*6*	*0,06*
$'G'$	*2*	*0,02*	$'T'$	*6*	*0,06*
$'H'$	*2*	*0,02*	$'U'$	*6*	*0,06*
$'I'$	*8*	*0,08*	$'V'$	*2*	*0,02*
$'J'$	*1*	*0,01*	$'W'$	*1*	*0,01*
$'K'$	*1*	*0,01*	$'X'$	*1*	*0,01*
$'L'$	*5*	*0,05*	$'Y'$	*1*	*0,01*
$'M'$	*3*	*0,03*	$'Z'$	*1*	*0,01*

Il y a, au total, cent jetons dont quinze marqués de la lettre E et un seul marqué de la lettre Z. La probabilité de la lettre E est alors de $\frac{15}{100} = 0,15$ *et celle de la lettre Z est de* $\frac{1}{100} = 0,01$.

Toutes les probabilités sont comprises entre 0 et 1, leur somme vaut 1 et elles sont formellement notées :

$$P(X = x) \forall x \in A$$

Elles expriment la probabilité que l'événement x, qui appartient à l'alphabet A, se réalise. Avec l'exemple précédent, on a $P(X =' E') = 0,15$ et $P(X =' W') = 0,01$. L'ensemble des probabilités de tous les événements x de A est appelé la *distribution* de la variable X. Cette notation, et en particulier la variable X, sera vue plus précisément par la suite.

Si, pour des raisons de délai ou de place requise, l'émetteur ne peut pas transmettre la source originale, alors il doit la compresser, c'est-à-dire à en réduire la taille. Cette *compression* peut être faite de manière à pouvoir *reconstruire* le message original soit à l'identique soit en autorisant des pertes.

La théorie de l'information, décrite au paragraphe 3.2.1, fournit le cadre d'étude pour estimer si une technique de compression fait subir des pertes au message source.

3.2.1. *Information et entropie*

Claude Shannon, ingénieur chercheur en télécommunication, est l'un des principaux initiateurs de la *théorie de l'information*. Cette théorie a pour objectif de mesurer la quantité d'informations émises par un message. Mais qu'est-ce qu'une information ?

En termes de télécommunication, un message qui n'est ni un bruit ni un signal purement aléatoire, contient une information. Cette information peut être perturbée par des redondances ou bien par des effets extérieurs indésirables, appelés bruits, comme la perte ou l'interférence du signal.

Dans ce contexte, mesurer la quantité d'informations revient alors à mesurer le nombre minimal de bits nécessaires au codage et à la quantification d'un signal sans en déformer l'information; il s'agit donc d'identifier et d'éliminer les redondances et le bruit du signal.

Ces travaux sont à l'origine de la théorie de l'information dont l'un des outils majeurs est l'*entropie*. Cette théorie est la base des deux processus décrits dans les sections suivantes : à savoir, la *quantification* (voir section 3.3) et le *codage entropique* (voir section 3.4). On va donc parcourir cette théorie pour en extraire les définitions et les théorèmes primordiaux pour la suite. Il ne s'agit pas d'expliquer *rigoureusement* et complètement (avec toutes les démonstrations souhaitables) la théorie de l'information, mais, d'en donner un aperçu. Le lecteur intéressé par ce vaste sujet, pourra se référer aux sites Internet de Yann Ollivier et de Louis Wehenkel[5], et à l'ouvrage de David S. Taubman et Michael W. Marcellin [TAU 02].

5. Leurs sites peuvent être facilement retrouvés à l'aide d'un moteur de recherche et de leurs noms.

3.2.1.1. *Définitions*

Pour parler de théorie de l'information, on va d'abord définir ce qu'est un message et l'information qu'il contient.

Soit un message X défini comme une suite de symboles prenant valeurs dans l'alphabet A. Quand l'ordre d'apparition des symboles n'est pas connu, le message X est modélisé par des *variables aléatoires*[6] (VA) :

$$X = (X_0, \cdots, X_j, \cdots)$$

Chaque VA X_j, prend pour valeur un *événement* x_i. L'ensemble des événements possibles est l'*alphabet* A. Dans le cas d'une VA discrète, l'alphabet est fini ou infini dénombrable :

$$A = \{x_0, x_1, \cdots, x_k, \cdots\}$$

La probabilté pour que la VA X_j soit effectivement l'événement x_i, est définie par sa *distribution* :

$$\{P(X_j = x_i); \forall x_i \in A\}$$

EXEMPLE 3.2.– *Soit, un alphabet A discret, constitué des deux mots* OK *et* KO. *Ces deux mots peuvent être représentés – plus tard, on dira codés – à l'aide d'un unique bit. Un message X, est donc constitué de VA discrètes X_j, qui peuvent se réaliser :*

– en l'événement OK *avec la probabilité* $P(X_j = \text{OK})$ *;*

– en l'événement KO *avec la probabilité* $P(X_j = \text{KO})$.

DÉFINITION 3.1.– *Quand, pour toute VA X_j, les événements réalisables ont tous la même chance d'apparaître – et ce quels que soient les événements passés ou à venir – alors ces événements sont dits équiprobables.*

Soit un alphabet discret $A = \{x_0, x_1, \cdots, x_k\}$, comportant k événements équiprobables, tel que $k = |A| = 2^n$. On a alors besoin de $n = \log_2 |A|$ bits pour coder un événement x_i. Cette valeur est la première mesure, appelée *information*[7], de l'événement x_i :

$$I_A(x_i) = \log_2 |A|$$

6. Une fois encore, on utilise une définition intuitive et approximative des variables aléatoires. On invite le lecteur intéressé à consulter le livre de Gérard Calot [CAL 67].

7. Aussi appelée *information propre*. Pour être rigoureux, on devrait parler de *quantité d'informations propres* au lieu d'information propre pour lever toute ambiguïté avec la notion sémantique d'information. Toutefois, la sémantique n'ayant pas sa place dans le cadre de la théorie de l'information, cette ambiguïté n'existe pas et le raccourci est toléré.

Mais, qu'en est-il de la probabilité qu'un *événement x appartient à un sous-ensemble $F \subseteq A$* ?

Pour répondre à cette question, il faut procéder en deux étapes successives. Premièrement, il faut spécifier la probabilité que x est un événement de F. Puis, il faut péciser de quel événement de F il s'agit.

De la même manière qu'il faut $\log_2 |A|$ bits pour coder x sachant qu'il appartient à l'alphabet A, on peut coder cet événement avec $\log_2 |F|$ bits s'il provient du sous-ensemble F. Il en découle que $(\log_2 |A| - \log_2 |F|)$ bits sont utilisés pour préciser que l'événement appartient au sous-ensemble F. On en déduit une mesure d'information :

$$I_A(F) = \log_2 |A| - \log_2 |F| = \log_2 \frac{|A|}{|F|}$$

Sachant que le rapport $\frac{|F|}{|A|}$ représente la fréquence d'apparition du sous-ensemble F, c'est-à-dire *sa probabilité* $p(F)$, l'information s'écrit aussi :

$$I_A(F) = \log_2 \left(\frac{1}{p(F)} \right) = -\log_2(p(F))$$

Et, en particulier, si $F = \{x\}$:

$$I_A(X = x) = I_A(\{x\}) = -\log_2(p(X = x))$$

La mesure d'information d'un événement est inversement proportionnelle au logarithme de sa probabilité.

Le graphe de la figure 3.11 en donne une explication intuitive. Plus un événement x, est probable – c'est-à-dire plus la VA X a la chance de devenir l'événement x – moins il est intéressant. (On pourrait dire qu'il est fatiguant de voir *toujours* la même information !) En revanche, lorsqu'un événement se fait rare – qu'il a une faible probabilité – l'information qu'il transporte doit être enregistrée dès que cet évènement apparaît.

REMARQUE 3.2.– *Il ne faut pas oublier que ce propos porte sur la quantité d'informations et non sur la sémantique de cette information. Sous cet éclairage, il est donc évident que plus un événement est rare, plus il est important de le mémoriser.*

L'*information moyenne* de l'ensemble A vaut :

$$\begin{aligned} H(A) &= \sum_{x \in A} p(x) I_A(X = x) \\ &= \sum_{x \in A} p(x) \left(-\log_2(p(x)) \right) \\ &= -\sum_{x \in A} p(x) \log_2(p(x)) \end{aligned}$$

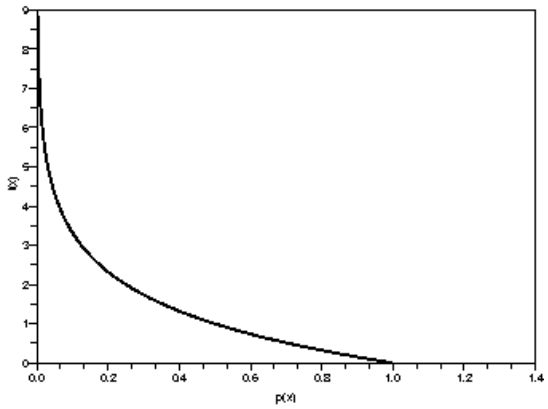


Figure 3.11. *Graphe de l'information d'un événement*

Cette information moyenne n'est autre que l'*entropie* de l'ensemble. C'est une mesure d'instabilité moyenne apportée par tous les événements associés à A. L'entropie est maximale quand tous les événements sont supposés équiprobables; autrement dit, quand l'instabilité est maximale. Une entropie maximale implique que le récepteur ne peut pas prédire ce qu'il va recevoir. Inversement, si un événement a une probabilité de 1, alors aucune information n'est fournie et l'entropie est nulle. Dans le cadre de la télécommunication, cela signifie qu'un récepteur, connaissant les probabilités des événements, n'a pas besoin de recevoir l'information pour la connaître !

THÉORÈME 3.3.– *Une mesure d'information sur n événements est une fonction d'entropie :*

$$H_n(X) = -\sum_{i=0}^{n-1} p_i \log_2(p_i)$$

où X est une VA d'alphabet $A = \{x_0, \cdots, x_{n-1}\}$ de probabilités $\{p_0, \cdots, p_{n-1}\}$ avec $p_i = P(X = x_i)$. Cette fonction vérifie les propriétés suivantes :

– $H_2(X) = 1$ *si les deux événements sont équiprobables :*

$$p_0 = p_1 = 1/2$$

– $H_n(X) \leqslant \log_2(|A|)$.

EXEMPLE 3.3.– *On choisit un exemple graphiquement représentable avec un univers décrit par une VA X d'alphabet $A = \{x_0, x_1\}$. Si la probabilité de x_0 vaut p, alors celle de x_1 vaut $(1-p)$ et l'entropie ne dépend que du seul paramètre p :*

$$H(\{x_0, x_1\}) = -p \log_2(p) - (1-p) \log_2(1-p)$$

La figure 3.12, montre le graphe $H(A)$. L'entropie est maximale quand :

$$p = 1 - p = \frac{1}{2}$$

C'est-à-dire, quand l'instabilité est maximale; il n'y a pas de prédiction des événements possibles. Elle est minimale quand p tend vers 0 (resp. 1). L'événement x_1 (resp. x_0) a beaucoup plus de chance d'être réalisé que l'autre.

THÉORÈME 3.4.– *Soit la suite finie de VA :*

$$S = \{X_0, X_1, \ldots, X_n\}$$

Son entropie est bornée supérieurement :

$$H(S) = H(X_0, X_1, \ldots, X_n) \leqslant H(X_0) + H(X_1) + \cdots + H(X_n)$$

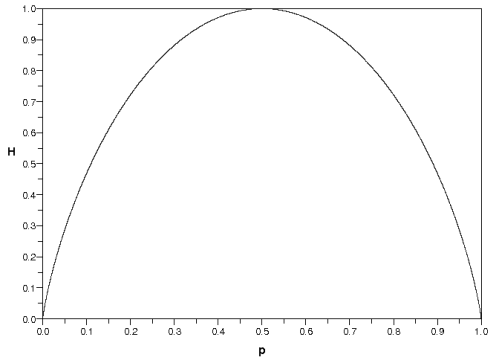


Figure 3.12. *L'entropie d'un système à deux événements*

REMARQUE 3.3.– *L'annexe B.5 fournit les définitions complémentaires nécessaires à l'établisement de ce théorème.*

DÉFINITION 3.2.– *Un processus aléatoire est une suite infinie de VA :*

$$X = \{X_0, X_1, \ldots, X_n, \ldots\}$$

Son entropie vaut :

$$H(X) = \lim_{n \to \infty} \frac{1}{n} H(X_0, \cdots, X_n)$$

quand cette limite existe !

EXEMPLE 3.4.– *Lors d'une communication entre un émetteur et un récepteur, un symbole émis au temps t est représenté par la VA X_t. La communication est alors une suite infinie de VA, $X = (X_0, \cdots, X_n, \cdots)$ indicée sur le temps.*

COROLLAIRE 3.1.– *Lorsque les VA sont indépendantes et identiquement distribuées*[8] *(IID), l'entropie du processus aléatoire vaut :*

$$\begin{aligned} H(X) &= \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} H(X_i) \\ &= \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} H(X_0) \\ &= H(X_0) \end{aligned}$$

3.2.1.2. *Information et codage*

Dans un théorème fondamental dont on ne fera pas la démonstration [TAU 02], Claude Shannon ajoute une relation entre l'entropie d'une source décrite par des VA IID et le codage de cette source.

THÉORÈME 3.5.– *Soit :*

– un processus aléatoire discret, $X = \{X_n\}$, de VA IID, d'entropie finie, $H(X)$, défini sur un alphabet discret $A = \{x_0, x_1, \ldots, x_{n-1}\}$;

– un codage de longueur fixe (m,L) qui associe un unique code de L bits à chaque vecteur $x = (x_0, x_1, \cdots, x_{m-1})$.

Alors la probabilité d'erreur $P_e(m, L)$, qu'un vecteur x ne soit pas identifiable par son code binaire, tend vers 0 quand m tend vers l'infini tant que :

$$\frac{L}{m} \geqslant H(X)$$

Et, réciproquement, $\lim_{m\to\infty} P_e(m, L) = 1$, tant que :

$$\frac{L}{m} < H(X)$$

En d'autres termes, lors du codage, il est primordial que le *débit binaire*, $R = L/m$, ne descende pas en deçà de la limite qu'est l'entropie du processus aléatoire. Les algorithmes de codage respectant ce critère sont dits *entropiques* et opèrent sans perte. En revanche, ceux qui ont un débit binaire inférieur à l'entropie effectuent un codage avec des risques de perte.

8. C'est-à-dire quand les VA sont définies sur le même alphabet avec la même distribution.

Plus précisément, une codification de chaque événement x_i à l'aide d'un nombre fixe L de bits ne permet pas de coder n'importe quelle source tout en garantissant que le débit binaire soit au-dessus de l'entropie de la source. Ainsi, est-il préférable d'utiliser des *codages à longueurs variables* de bits. C'est le cas des algorithmes décrits en section 3.4.

Ce théorème fait référence à des VA IID. Les processus aléatoires, décrits par de telles VA, représentent des sources dites *sans mémoire*. Lorsque la source est *avec mémoire*, la VA au temps t, notée X_t, dépend de p VA la précédant dans le temps. La probabilité que X_t soit x_t est alors conditionnée par ces p précédentes VA :

$$P(X_t = x_t | X_{t-p} = x_{t-p}, \cdots, X_{t-1} = x_{t-1})$$

Les processus aléatoires représentant ces sources sont dits de *Markov*. Le paramètre p est l'ordre de ces processus. Le théorème précédent a été étendu à de tels processus. Les algorithmes de codage qui en découlent ne seront pas présentés puisqu'ils ne sont pas utilisés par la suite. A nouveau, le lecteur soucieux de poursuivre cette étude peut se référer à [TAU 02, BAR 02].

3.2.1.3. *Information et quantification*

Quantifier un processus aléatoire $X = \{X_i\}$, consiste à transformer les événements x_i, provenant d'un alphabet A, en un nombre *fini* d'événements $\hat{x}$, provenant d'un alphabet $\hat{A}$, tel que :

$$|\hat{A}| < |A|$$

Dans le cadre de la quantification, les événements x_i sont regroupés en vecteurs $x_k = (x_{k,0}, \cdots, x_{k,m-1})$ et les événements $\hat{x}_j$ en vecteurs $\hat{x}_l = (\hat{x}_{l,0}, \cdots, \hat{x}_{l,m-1})$. Puisque les événements $\hat{x}_j$ sont en nombre fini, les vecteurs $\hat{x}_l$ le sont aussi. Ils sont rassemblés en un *dictionnaire*[9], $\mathcal{B} = \{\hat{x}_0, \cdots, \hat{x}_{B-1}\}$, de taille B.

REMARQUE 3.4.– *Du fait que l'alphabet $\hat{A}$ est réduit par rapport à l'alphabet A, le dictionnaire $\mathcal{B}$ fournit une même valeur pour des événements source différents. Il y a donc obligatoirement des pertes qu'il faut gérer au mieux.*

La *quantification* revient alors à trouver l'indice dans le dictionnaire, $\mathcal{B}$, qui minimise une certaine mesure de distorsion, ρ_m, entre les vecteurs x_k et $\widehat{x}_l$:

$$l^* = Q(x_k) = \underset{l \in \{0, \cdots, B-1\}}{\operatorname{argmin}} \rho_m(x_k, \hat{x}_l)$$

9. Aussi appelé *codebook*.

La *déquantification* est alors l'opération permettant de retrouver la valeur quantifiée connaissant son indice dans le dictionnaire :

$$Q^{-1}(l^*) = \hat{x}_{l^*}$$

Bien qu'il soit courant dans la littérature de définir la quantification comme la composition de ces deux applications :

$$\hat{x}_{l^*} = Q^{-1}(Q(x_k))$$

REMARQUE 3.5.– *On garde la version séparée faisant intervenir les indices. Ce choix se justifie par le fait qu'en pratique (en dehors de certains cas particuliers comme celui du CAN) l'émetteur communique l'indice au récepteur et non la valeur quantifiée. Il est ainsi supposé que le récepteur et l'émetteur connaissent le dictionnaire. D'un point de vue théorique, il n'y a pas de différence entre les deux versions*[10]*. La figure 3.13 illustre le schéma général de la quantification.*

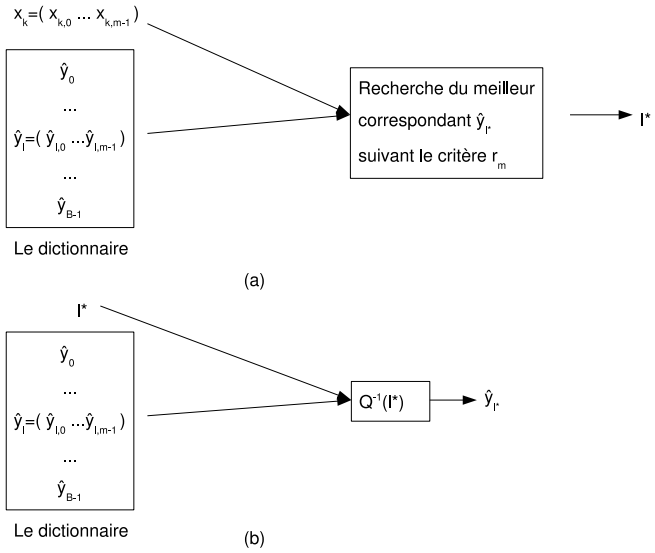


Figure 3.13. *Schéma général : (a) de la quantification $Q(x_k)$ (b) de la déquantification $Q^{-1}(l^*)$*

10. En pratique, il y a des avantages et des inconvénients dans les deux approches. La version indicée a généralement un débit binaire plus faible, mais, elle demande à ce que l'émetteur et le récepteur se synchronisent sur le dictionnaire à utiliser.

Habituellement, la mesure de distorsion ρ_m, est la somme des distorsions mesurées sur chacune des dimensions des vecteurs :

$$\rho_m(x, \hat{x}) = \frac{1}{m} \sum_{i=0}^{m-1} \rho(x_i, \hat{x}_i)$$

où ρ est une fonction positive de mesure de distorsion entre événements.

Muni de cette mesure de distorsion, on peut mesurer la distorsion moyenne du dictionnaire :

$$D_{\mathcal{B}} = E[\rho_m(X, \hat{X})]$$

où $X = (X_0, X_1, \cdots, X_{m-1})$ et $\hat{X} = (\hat{X}_0, \hat{X}_1, \cdots, \hat{X}_{m-1})$.

3.2.1.3.1. Débit binaire

On peut formuler la taille B, du dictionnaire en fonction de la taille, m, des vecteurs x_k :

$$B = 2^{Rm} \text{ avec } R > 0$$

Le *débit binaire* R, indique combien de bits sont utilisés pour stocker ou transmettre un événement. Il correspond au rapport entre la taille du vecteur $\hat{x}_l$ et celle du vecteur x_k :

$$R = \frac{|\hat{x}_l|}{|x_k|} = \frac{\log_2(B)}{m}$$

Ce débit binaire est déjà utilisé dans le théorème 3.5 (page 85) à propos du codage entropique. Il permet également de définir le *taux de compression* du dictionnaire :

$$T = \frac{\log_2 |A|}{R} = m \frac{\log_2 |A|}{\log_2 |\mathcal{B}|}$$

3.2.1.3.2. Relations entre débit binaire et distorsion

DÉFINITION 3.3.– *L'abaque $R(D)$, du débit binaire en fonction de la distorsion correspond à l'information minimale $H(\hat{X})$ requise pour coder la valeur quantifiée de X telle que la distorsion du dictionaire $D_{\mathcal{B}}$, soit inférieure à la valeur D :*

$$R(D) = \min H(\hat{X}) \textit{ sous la contrainte } D_{\mathcal{B}} < D$$

En émettant certaines hypothèses [TAU 02, BAR 02, GUI 96] – entre autres celle d'une distribution gaussienne – l'abaque de la distorsion en fonction du débit binaire est de la forme :

$$D(R) \cong 2^{-2R}$$

L'hypothèse de distribution gaussienne est communément acceptée car elle reflette souvent la réalité lorsque les données sont en grand nombre. De plus, elle est une borne supérieure à $D(R)$ [TAU 02]. La courbe est donc convexe et sert de référence pour construire des algorithmes de quantification et de codage.

La figure 3.14 montre l'allure générale de cette fonction. Naturellement, la distorsion est maximale quand le débit binaire est nul, et elle diminue au fur et à mesure que le débit binaire augmente. Comme l'indique le théorème 3.5, la distorsion est nulle dès que l'entropie de la source est atteinte.

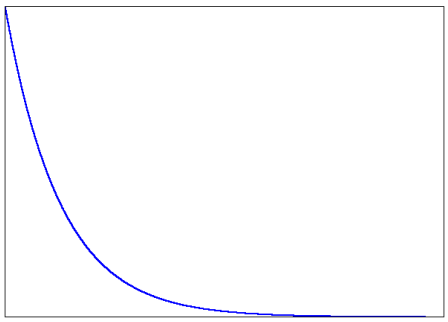


Figure 3.14. *L'allure de la distorsion en fonction du débit binaire*

Cet abaque fournit une référence dans le cadre de la quantification, mais, également dans le cadre du codage. Elle est utilisée par JPEG2000 et MPEG4-AVC.

3.3. Quantification

La quantification engendre toujours des pertes. Elle intervient à différents niveaux avec des objectifs différents. Les pertes ne sont donc pas les mêmes suivant l'application visée. En pratique, il existe deux catégories de quantificateurs.

La première catégorie est celle des quantificateurs utilisés lors de la conversion d'un signal analogique en sa version numérique par un CAN. Dans ce cas, la quantification a pour objet de fixer la valeur discrète la plus proche possible de la valeur analogique. Typiquement, il s'agit de convertir des volts en nombres entiers ou réels *machines*. La quantification est de type scalaire pour des raisons de facilité d'implémentation, d'efficacité en temps de calcul et de précision. Cette catégorie a déjà été évoquée dans la section 3.1.

Contrairement à ceux de la première catégorie, les quantificateurs de la deuxième catégorie autorisent des pertes perceptibles. Ils servent à la compression du signal pour en faciliter le stockage et le transport. On les retrouvera dans les prochains chapitres.

La quantification peut être scalaire ou vectorielle. En pratique les quantificateurs scalaires sont préférés pour leur facilité d'implémentation, mais, aussi parce qu'associés aux techniques de codage entropique, ils se révèlent être très efficaces.

Le schéma général de la quantification, donné au paragraphe 3.2.1.3, est vectoriel. La quantification scalaire n'est donc qu'un cas particulier de la quantification vectorielle. Toutefois, elle est présentée séparément de la quantification vectorielle, pour en simplifier la compréhension.

3.3.1. *Quantification scalaire*

La quantification scalaire, Q, met en relation un alphabet A, discret ou continu, borné ou non, de valeurs réelles x_i avec un ensemble $I_{\mathcal{B}}$ discret fini de valeurs entières. Les vecteurs x_k, définis au paragraphe 3.2.1.3 sont donc monodimensionnels : $m = 1$.

Les valeurs j de $I_{\mathcal{B}}$ sont les indices du dictionnaire $\mathcal{B} = \{\hat{x}_0 \cdots \hat{x}_{B-1}\}$ qui n'est autre que l'alphabet $\hat{A}$, lui-même :

$$Q(x_i) = j \text{ et } Q^{-1}(j) = \hat{x}_j$$

Le modèle de la quantification correspond à un découpage de l'alphabet A en régions R_j avec $0 \leqslant j < B$. Tous les x_i d'une région R_j font référence à l'indice j :

$$Q(x_i) = j$$

et ils ont pour *représentant* $\hat{x}_j$:

$$Q^{-1}(j) = \hat{x}_j \text{ avec } (\hat{x}_j \in \mathcal{B})$$

La figure 3.15 illustre ce découpage en régions R_j et leurs représentants.

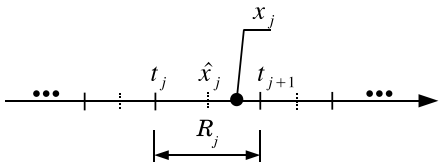


Figure 3.15. *Partitionnement de l'axe des réels en régions R_j de représentants les valeurs $\hat{x}_j$*

Ce découpage doit respecter la distribution des valeurs de A, si celle-ci est connue. Les intervalles d'une quantification optimale sont plus resserrés pour les fortes probabilités que pour les faibles (voir figure 3.16). Le quantificateur sera uniforme uniquement si la distribution l'est également !

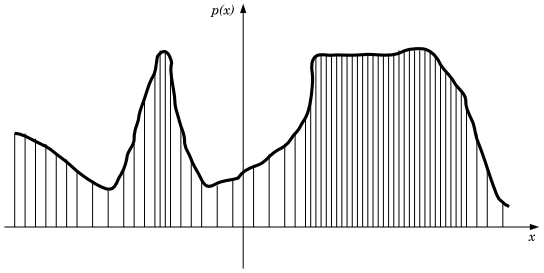


Figure 3.16. *Les pas de quantification s'adapte à la distribution*

REMARQUE 3.6.– *Classiquement, l'ensemble A est centré : sa valeur moyenne vaut zéro. Au besoin, les valeurs sont translatées lors de la quantification. La translation inverse est faite par la déquantification.*

La quantification scalaire peut être *paire* ou *impaire* suivant que la valeur nulle est une frontière entre deux régions ou pas.

Elle est *uniforme* si le pas de quantification Δ, définissant la taille des régions de A, est constant. Quand le pas Δ, n'est pas constant, la quantification est nonuniforme.

La figure 3.17 illustre ces définitions. En pratique, A est évidemment borné.

3.3.1.1. *Algorithme de LLoyd-Max*

L'algorithme de LLoyd-Max est une référence parmi les algorithmes de quantification optimale. Le découpage est effectué en un nombre fixe B, d'intervalles :

$$I_k = [t_k, t_{k+1}[\text{ avec } k = 0, 1, \cdots, B-1$$

L'algorithme de LLoyd-Max définit les seuils t_k, et les représentants $\hat{x}_k$ des régions R_k tel que la distorsion induite soit minimale. La distorsion est estimée aux moindres carrés :

$$\begin{aligned} D_{\mathcal{B}} &= E[(X-\hat{X})^2] \\ &= \sum_{k=0}^{B-1} \int_{t_k}^{t_{k+1}} P(X=x)(x-\hat{x}_k)^2 \mathrm{d}x \\ &= \cdots \int_{t_{k-1}}^{t_k} P(X=x)(x-\hat{x}_{k-1})^2 \mathrm{d}x + \int_{t_k}^{t_{k+1}} P(X=x)(x-\hat{x}_k)^2 \mathrm{d}x + \cdots \end{aligned}$$

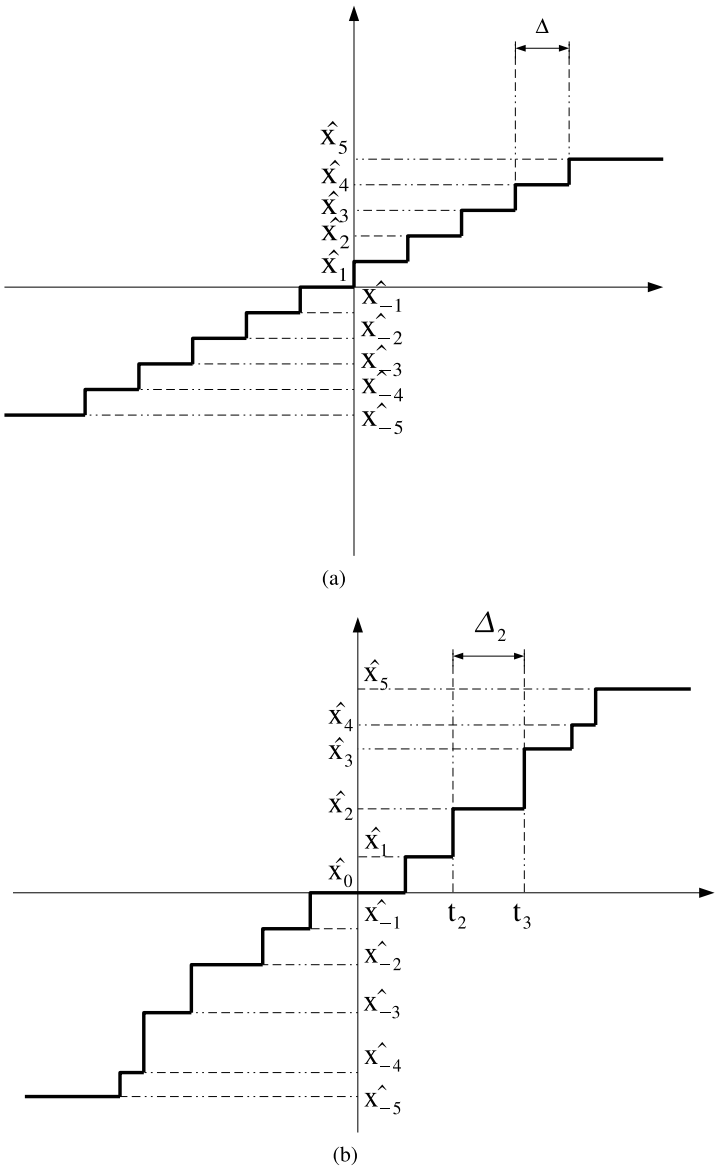


Figure 3.17. *Quantificateur scalaire (a) pair uniforme (b) impair non uniforme*

Minimiser cette distorsion revient à estimer les seuils t_k et les représentants $\hat{x}_k$ des intervalles I_k; c'est-à-dire, à annuler les dérivées suivant ces deux paramètres :

$$\begin{aligned} \frac{\partial D_{\mathfrak{B}}}{\partial t_k} &= P(X=t_k)(t_k-\hat{x}_{k-1})^2 - P(X=t_k)(t_k-\hat{x}_k)^2 = 0 \\ \frac{\partial D_{\mathfrak{B}}}{\partial \hat{x}_k} &= -2\int_{t_k}^{t_{k+1}} P(X=x)(x-\hat{x}_k)\mathrm{d}x = 0 \end{aligned}$$

qui donne :

$$t_k = \frac{\hat{x}_{k-1}+\hat{x}_k}{2} \tag{3.2}$$

$$\hat{x}_k = \frac{\int_{t_k}^{t_{k+1}} xp(x)\mathrm{d}x}{\int_{t_k}^{t_{k+1}} p(x)\mathrm{d}x} \tag{3.3}$$

L'équation (3.2) signifie que le seuil t_k se trouve être le milieu des deux représentants $\hat{x}_k$ et $\hat{x}_{k+1}$. L'équation (3.3) signifie que le représentant $\hat{x}_k$ est le barycentre du nuage de points de l'intervalle I_k.

Sachant comment calculer les représentants et les seuils des intervalles de quantification, il existe un algorithme itérant sur ces paramètres jusqu'à obtenir un découpage optimal :

LLOYD-MAX()
1 initialiser le dictionnaire $\mathfrak{B}_1$
2 $m \leftarrow 1$
3 **répéter**
4 calculer les t_k suivant l'équation 3.2
5 calculer les $\hat{x}_k$ suivant l'équation 3.3
6 $m \leftarrow m+1$
7 **jusqu'à** $\frac{D_{\mathfrak{B}_{m-1}} - D_{\mathfrak{B}_m}}{D_{\mathfrak{B}_m}} < \epsilon$ // ϵ *est un paramètre pré-établi*

3.3.1.2. *Quelles améliorations possibles ?*

Cet algorithme n'est optimal que dans certaines conditions [TAU 02]. Aussi existe-t-il des variantes où plus de latitude est donnée aux calculs des seuils des intervalles.

Une de ces versions ne calcule pas les bornes t_k. Elle utilise l'algorithme du plus proche voisin pour connaître le représentant $\hat{x}_k$ de chaque valeur x_i de A. Les intervalles sont alors de la forme :

$$I_k = \{x_i \in A \text{ t.q. } (x_i-\hat{x}_k)^2 < (x_i-\hat{x}_q)^2 \forall q \neq k\} \text{ avec } k,q = 0,1,\cdots,B-1$$

Cette version de l'algorithme suppose que l'alphabet A est dénombrable; ce qui est le cas dans la pratique. En revanche, il est supposé que toutes les données de A sont connues *a priori* ; ce qui est une contrainte forte.

Il existe une autre version [TAU 02] qui relâche cette dernière contrainte. Seule une partie des données de A est connue. Mais, en termes de distribution, cette partie est supposée être représentative de l'ensemble des données de A.

D'autres techniques, appelées *quantifications entropiques*, visent à minimiser la distorsion $D_{\mathcal{B}}$ sous la contrainte que le débit binaire R est supérieurement proche de l'entropie de la source X. On ne détaillera pas plus ces techniques qui, parce qu'elles sont plus lourdes à mettre en œuvre, ne sont pas utilisées dans le cadre du multimédia.

3.3.1.3. *Quelle relation entre codage et quantification scalaire ?*

Sachant que le nombre L, de bits vaut $\log_2(B)$ et que la quantification scalaire porte sur des vecteurs monodimensionnels ($m = 1$), le débit binaire R, est borné supérieurement :

$$R \leqslant \frac{\log_2(B)}{m} = \frac{L}{m} = L \text{ (quand } m = 1)$$

Cette valeur est valide en pratique si $\log_2(B)$ est une valeur entière. Sinon, il faut trouver une valeur entière approchante :

$$R \leqslant \lceil \log_2(B) \rceil \text{ est une valeur acceptable.}$$

Ainsi, pour une source $\{x_0, x_1, \ldots, x_n\}$, le codage à *longueur fixe* fournit un débit binaire constant :

$$|Q(x_i)| = L, \forall x_i \in A$$

Mais, pour s'approcher au plus près de la borne inférieure qu'est l'entropie de la source, la quantification est habituellement à *longueur variable* (voir paragraphe 3.4.3) :

$$H(\hat{X}) \lesssim R \leqslant \log_2(B)$$

Un nombre différent de bits est alors alloué à chacun des symboles de l'alphabet A :

$$|\hat{x}_i| = l_i \leqslant L, \forall \hat{x}_i \in \mathcal{B}$$

3.3.1.4. *Quantification uniforme*

Soient les conditions de quantification suivante :

– le nombre de niveaux de quantification est infini ;

– les pas de quantification sont de la forme $k\delta$ où δ est une constante et k parcourt l'ensemble des entiers.

Les intervalles sont alors :

$$I_k = \left[k\delta - \frac{\delta}{2}, k\delta + \frac{\delta}{2}\right[\text{ avec } k = \cdots, -2, -1, 0, 1, 2, \cdots$$

3.3.1.4.1. Quand utiliser la quantification uniforme ?

Pour des valeurs raisonnablement faibles[11] de δ, la distorsion vaut [BAR 02, TAU 02] :

$$D_{\mathcal{B}} \cong \frac{\delta^2}{12}$$

et l'équation de l'entropie des valeurs quantifiées se simplifie [BAR 02, TAU 02] :

$$H(\hat{X}) \cong H(X) - \log_2(\Delta)$$

Par ailleurs, si le codage entropique est efficace, le débit binaire est alors proche de la valeur d'entropie :

$$R \cong H(\hat{X})$$

On en déduit une estimation du pas de quantification :

$$\Delta \cong 2^{H(X)-R}$$

Ainsi, la distorsion s'écrit en fonction du débit binaire :

$$D(R) \cong \frac{1}{12}\left(\frac{2^{2H(X)}}{2^{2R}}\right) \geqslant 2^{-2R}$$

En conclusion, pour des débits binaires importants, la distorsion s'approche de la valeur optimale théorique de l'équation (3.3). Les quantificateurs uniformes sont alors *quasi* optimaux. Pour les faibles débits, le quantificateur uniforme reste, malgré tout, une approximation correcte [TAU 02].

11. Assez faible pour que la distribution au sein de chaque intervalle soit correctement approximée comme uniforme.

3.3.1.4.2. Quantification uniforme à zone morte

Afin de mieux approcher l'abaque $D(R)$ théorique, l'intervalle I_0 peut être élargi. Les quantificateurs adoptant cette démarche sont dits à *zone morte*. Classiquement, l'intervalle I_0 est de largeur double des autres intervalles. Le fait de doubler la largeur de l'intervalle central augmente la distorsion. Mais, en même temps, l'entropie diminue, ce qui a un effet compensatoire. Les intervalles d'un quantificateur à zone morte sont de la forme (voir figure 3.18) :

$$I_k = \left\{ \begin{array}{ll} [-\delta, +\delta[& \text{si } k = 0 \\ [k\delta, (k+1)\delta[& \text{si } k > 0 \\ [(k-1)\delta, k\delta[& \text{si } k < 0 \end{array} \right\}$$

Le quantificateur est des plus simples :

$$Q(x) = k = \text{sign}(x) \lfloor \frac{|x|}{\delta} \rfloor$$

Le déquantificateur est également simple :

$$\hat{x}_k = \text{sign}(k) \left(|k| + \frac{1}{2} \right) \Delta$$

Le fait de décomposer l'indice k, en son signe $\text{sign}(k)$, et son module $|k|$, permet d'écrire des algorithmes efficaces.

Figure 3.18. *Le quantificateur uniforme à zone morte*

3.3.1.4.3. Quantification et multirésolution

En supposant que le module $|k|$ soit constitué de q bits de valeur :

$$k = sb_0b_1 \cdots b_{q-1}$$

avec s le bit de signe et b_0 le bit de poids fort.

Ignorer les p bits de poids faible, $k^{(p)} = sb_0 \cdots b_{q-1-p}$, revient à quantifier avec un pas de $2^p\Delta$:

$$k^{(p)} = Q^{(q,p)} = \text{sign}(s) \lfloor \frac{|x|}{2^p\Delta} \rfloor$$

Le format JPEG2000 utilise ce schéma de quantification pour les coefficients d'ondelettes de pas dyadique (voir chapitre 2). Ceci permet d'intégrer l'approche multirésolution de la décomposition en ondelettes au sein du quantificateur.

3.3.1.5. *Quantification matricielle*

Quand, en termes de perception humaine, certaines propriétés de la source sont connues, il est possible d'envisager une solution plus heuristique reposant sur la construction d'un tableau de pondérations associé aux intervalles de la quantification. A chaque intervalle, est fixée une valeur qui viendra quantifier au mieux la plage des données source concernées. L'ensemble des valeurs est rangé dans un tableau qui est, généralement, construit de manière expérimentale. Les quantificateurs et les déquantificateurs doivent alors connaître les mêmes tableaux.

Par exemple, une image peut être décrite par ses fréquences. Le chapitre 2 montre que les hautes fréquences d'une image sont spatialement localisées; elles représentent soit du bruit, soit du détail. Aussi, les amplitudes de ces hautes fréquences peuvent être diminuées, voire éliminées, sans que la perception que l'utilisateur a de l'image en soit fortement perturbée.

Le procédé peut être binaire comme dans l'exemple 2.9 (page 42). Le masque est un tableau de la taille d'un bloc (8x8). Il indique par `1` la conservation de la fréquence et par un `0` son élimination. Mais ce masque est habituellement plus souple. Les valeurs binaires sont remplacées par des valeurs réelles (ou entières) qui atténuent plus ou moins les fréquences.

Les standards JPEG et MPEG utilisent cette quantification matricielle qui repose sur des études expérimentales dont le cadre est fixé par une commission internationale. Ce cadre expérimental a été imposé afin de permettre une interopérabilité complète entre les quantificateurs et les déquantificateurs des différents industriels. Suivant ce procédé, le quantificateur et le déquantificateur font simplement référence aux valeurs de ce tableau. Une illustration en est donnée par l'exemple dans la section quantification du chapitre JPEG du volume 2.

3.3.1.6. *Quantification non uniforme*

Mais, la quantification uniforme n'est pas toujours adaptée. Lorsque le modèle de distribution de l'alphabet source est connu *a priori*, il est possible de construire un quantificateur optimal pour des données non uniformes.

3.3.1.6.1. Quantification par compansion

Comme expliqué dans le chapitre 4, l'oreille humaine a une sensibilité, de forme logarithmique, comprise entre 20 Hz et 20 kHz. La quantification nonuniforme s'impose pour approcher au mieux la forme logarithmique de cette sensibilité.

La solution qui a été envisagée – et qui est utilisée depuis longtemps dans le domaine de la télécommunication – est de transformer cette distribution de type logarithmique en une distribution uniforme. Cette transformation permet d'utiliser un quantificateur uniforme qui est alors optimal. Les lois de transformation utilisées doivent évidemment être réversibles afin d'appliquer la transformation inverse sur les valeurs quantifiées au niveau du récepteur.

Le schéma général de la figure 3.19 est appelé *compansion* qui est la contraction de *compression/expansion*[12]. Dans le cadre de la transmission de la parole, les lois de compansion sont la μ-loi et la loi *A* définies au paragraphe 4.2.3 page 152.

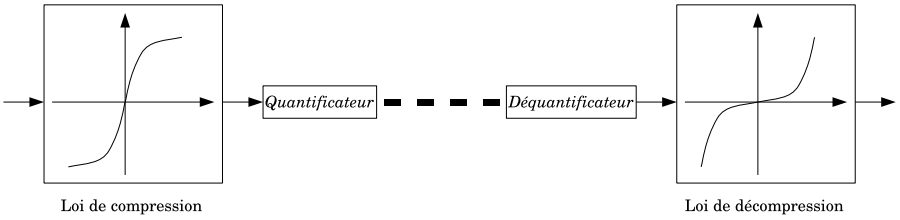


Figure 3.19. *La quantification par compansion*

3.3.2. *Quantification vectorielle*

La quantification scalaire uniforme, étudiée au paragraphe 3.3.1.4, considère les données sources comme indépendantes et uniformément distribuées. La section précédente apporte une réponse dans le cas de distributions non uniformes dont le modèle est connu. Mais, les données sont considérées indépendantes.

Dans les faits, cette situation est peu fréquente. Il est en effet plus courant que les données d'une source soient en relation entre elles. Par exemple, un pixel d'une image photographique n'est pas sans relation avec ces voisins; il est fort probable que son intensité soit proche de celle de certains de ses voisins.

La quantification vectorielle se propose de prendre en compte ces propriétés de *dépendance contextuelle*. A nouveau la construction de tels quantificateurs se fonde sur des expérimentations préétablies.

Il s'agit en fait de reprendre le cas général (voir paragraphe 3.2.1.3), jusqu'ici, limité au cas scalaire : maintenant $m > 1$. Les données et les représentants ne sont donc plus scalaires, mais, vectoriels. Chaque vecteur regroupe plusieurs échantillons

12. Traduction libre pour *companding* : *compressing/extanding*.

de la source. Les représentants doivent être enregistrés explicitement. Le dictionnaire[13] rassemblant ces représentants est connu du quantificateur et du déquantificateur. La figure 3.13 (page 87) du schéma général correspond parfaitement à la quantification vectorielle.

Par exemple, pour la compression d'une image, la quantification scalaire opère sur chaque pixel indépendamment. Mais, sachant que les pixels voisins sont fortement correllés, on a intérêt à les regrouper en blocs comme le montre la figure 3.20.

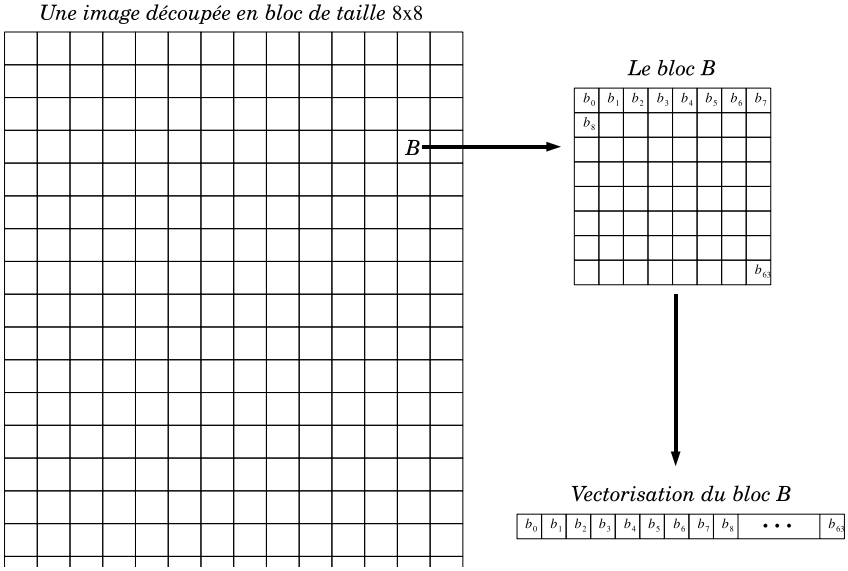


Figure 3.20. *Construction des vecteurs de données : découpage de l'image en blocs (de taille* 8×8*)*

Chaque bloc est transcrit sous forme de vecteur : les lignes du bloc sont mises bout à bout pour former un vecteur ligne. Ensuite, ce vecteur est comparé aux représentants du dictionnaire; ces représentants sont également des vecteurs. La recherche du meilleur représentant se fait en minimisant un critère préétabli : $\rho_m(\hat{x}, x)$. Les classes sont alors de la forme :

$$I_k = \{x \in A^m : \rho_m(x, \hat{x}_k) \leqslant \rho_m(x, \hat{x}_j) \forall j \in \{0, 1, \cdots, B-1\}\}$$

13. *Codebook*, par angliscime.

Les algorithmes théoriques et pratiques de recherche des seuils et des représentants de la quantification scalaire se généralisent naturellement.

Mais, la taille des vecteurs, ainsi que celle du dictionnaire, entraine très vite des chutes de performance. Pour éviter celà, le dictionnaire est construit comme un arbre dont la racine est le dictionnaire le plus général.

La valeur x à quantifier est tout d'abord présentée à ce dictionnaire racine. L'index résultant indique le numéro du nœud, fils de la racine, qui contient également un dictionnaire. Ce dictionnaire va approcher plus finement la valeur x. Le processus est itéré jusqu'à atteindre l'une des feuilles de l'arbre qui informe alors sur le représentant effectif de x.

La figure 3.21 illustre le procédé pour un dictionnaire binaire de profondeur 3 qui contient donc $2^3 = 8$ représentants effectifs. Seuls les indices des dictionnaires feuilles sont envoyés par l'émetteur; le récepteur parcourant l'arbre à l'aide de ceux-ci.

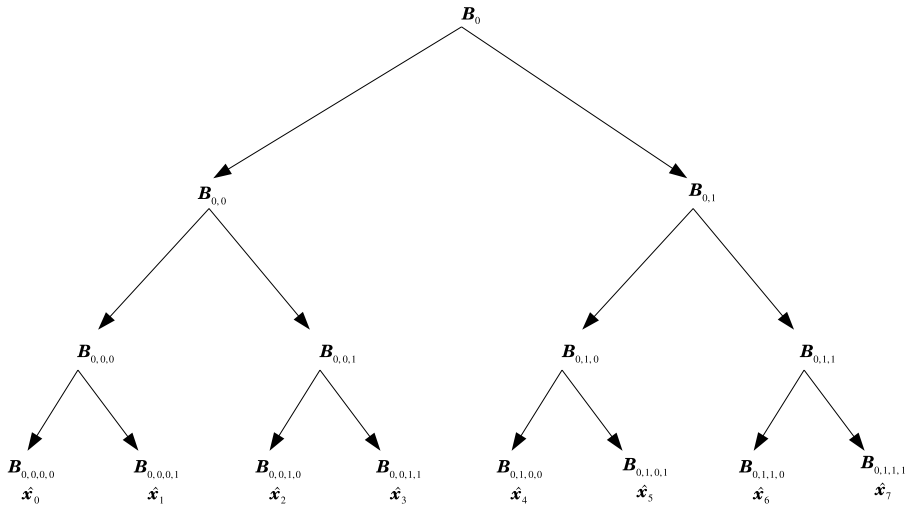


Figure 3.21. *Structure arborescente pour la quantification vectorielle*

3.3.3. *Synthèse de la quantification*

La quantification scalaire est préférée à la quantification vectorielle pour son efficacité en temps et sa simplicité algorithmique. De plus, le contexte qui informe sur les relations de dépendance entre les échantillons peut être facilement pris en compte à l'étape du codage. Ainsi, actuellement (en 2008), le couple <*quantification scalaire, codage contextuel*> est préféré à une quantification vectorielle complexe.

Une autre solution consiste à transformer le dictionnaire en une table de hachage. Chaque contexte est une clé pour la table de hachage; plusieurs contextes définissent évidemment la même clé ! Pour être efficace, la construction de la clé repose sur l'utilisation de tables de contextes déterminées expérimentalement.

Le lecteur intéressé pourra consulter [TAU 02, BAR 02]. Il pourra également y trouver d'autres formes de quantification, telles que la quantification par treillis.

3.4. Codage

La théorie de l'information stipule que tout codage de longueur fixe L est garanti sans perte si le débit binaire, $R = L/m$, reste supérieur ou égal à l'entropie de la source (voir théorème 3.5 page 85). Ceci limite l'utilisation des codeurs à longueur fixe à des familles de messages dont l'entropie maximale est connue *a priori*.

Pour coder efficacement une source sans en connaître au préalable l'entropie, il faut avoir recours à d'autres techniques. Les codeurs par plages (voir paragraphe 3.4.1), les codeurs à dictionnaires dynamiques (voir paragraphe 3.4.2), les codeurs à longueurs variables (voir paragraphe 3.4.3) et les codeurs arithmétiques (voir paragraphe 3.4.4) proposent diverses solutions afin de s'approcher de l'entropie de la source à coder, tout en garantissant de ne pas descendre en deçà.

3.4.1. *Codage par plages*

Le principe des codeurs par plages (RLC[14]) repose sur la constatation que certaines sources présentent des suites consécutives de données identiques. Par exemple, dans une image de synthèse, comme un logo, on perçoit plusieurs régions d'intensité et de couleur identiques. Il est donc intéressant de ne coder qu'une seule fois la valeur de la donnée et d'en compter le nombre d'occurrences consécutives. Cette technique, simple à mettre en œuvre, ne peut en aucun cas avoir un débit binaire en dessous de l'entropie de la source

L'algorithme de base remplace chaque suite consécutive par un quadruplet de la forme <*marqueur, valeur, nombre d'occurrences, marqueur*>. Ce marqueur pouvant être une donnée à coder, il est dupliqué dans la source avant de procéder au codage proprement dit.

EXEMPLE 3.5.– *Soit le marqueur : # ; et la source à coder :*
aaaaa#55555555555#ababc.

14. RLC : *Run Length Coder*; ou RLE : *Run Length Encoder*.

L'alphabet est le code ascii; donc des valeurs codées sur 8 bits. Ainsi, la source possède 23 *octets. Le prétraitement consiste à dupliquer le marqueur dans la source :*

aaaaa##55555555555##ababc

Puis, le codeur remplace les suites. Avec le même alphabet que celui de la source, ce code possède 17 *octets :*

#a5####511###ababc

Le décodage est immédiat, sachant que l'alphabet est le code ascii. La lecture des codes se fait octet par octet. A chaque lecture d'un octet, le décodeur vérifie s'il s'agit d'un marqueur. Si c'est le cas, il consulte l'octet suivant. Lorsque ce dernier est de nouveau le marqueur, il le supprime et émet le marqueur comme valeur décodé. Quand l'octet suivant n'est pas le marqueur, il sait qu'il est en présence d'un quadruplet dont le second octet est la valeur. Les octets suivants (mis à part le dernier qui indique la fin du quadruplet) indiquent le nombre d'occurrences.

On contate facilement que la gestion des marqueurs peut devenir contraignante voire affaiblir le taux de compression au point de le rendre inintéressant. Le choix du marqueur est donc important; il ne doit pas être trop fréquent dans la source. Par ailleurs, rien ne garantit que la source possède beaucoup de séquences et que celles-ci soient de longueurs justifiant l'utilisation d'un codeur par plages.

Cependant, cette technique a été développée par le CCITT dans le cadre de la transmission par télécopieur[15] d'images binaires. Le marqueur devient alors inutile puisque :

– les lignes de ces images sont une alternance de séquences de pixels blancs et de pixels noirs ;

– les standards du CCITT impose que toute ligne commence par une séquence de pixels blancs, quitte à indiquer que la première séquence est de longueur nulle.

Dans la version G3 du CCITT, le codage est de type RLC-1D. Les lignes sont codées indépendamment les unes des autres. Un marqueur de fin de ligne (EOL) est utilisé.

Ainsi, les séquences se suivent en alternance blanche-noire, en commençant par une séquence blanche et en se terminant par EOL. La taille maximale d'une séquence est fixée à $1\ 728$; c'est la taille maximale d'une ligne d'un fac-similé. L'alphabet des longueurs des séquences est donc le sous-ensemble des entiers compris entre 0 et $1\ 728$. L'alphabet, mémorisé au sein du codeur et du décodeur, est alors de taille conséquente. Pour limiter cet alphabet, les longueurs à coder T, sont modélisées comme suit :

$$T = (64 * Q) + R$$

15. *Par téléfax*, *par fax* sont des termes également utilisés.

Les valeurs entières $(64 * Q)$ et R sont codées suivant un codage de Huffman modifié (voir paragraphe 3.4.3.1 page 110) dont la principale caractéristique est de fournir un code minimal *préfixé* (voir paragraphe 3.4.3 page 107). La table du codage de Huffman modifié est partiellement donnée en tableau 3.1.

Dans la version CCITT G4, le codage est dit RLC-2D. Le codage d'une ligne se fait en référence à une ligne déjà codée. Seuls les changements relatifs à cette ligne de référence sont codés. Il existe trois modes de codage (voir figure 3.22) :

1) *Mode passant*. La figure 3.22a indique que le pixel q_1 est situé avant (plus à gauche que) le pixel p_1. Dans ce cas, le marqueur `0001` est envoyé et le prochain quintuplet commence à la position de q_1 : la prochaine position de p_o est celle de q_1, mais, en restant sur la ligne en cours de codage ;

2) *Mode vertical*. La figure 3.22b indique que la distance entre les pixels p_1 et q_0 est au plus de 3 (q_1 est après p_1). Dans ce cas, le code retourné est la valeur de cette distance algébrique qui exprime la position relative de p_1 par rapport à q_0. p_1 devient le point de référence p_0 du prochain quintuplet ;

3) *Mode horizontal*. La figure 3.22c indique une distance entre p_1 et q_0 supérieure à 3 pixels (q_1 est après p_1). Dans ce cas, le code retourné est le triplet constitué :

a) du marqueur `001`,

b) du codage RLC-1D de la séquence (p_0, p_1),

c) du codage RLC-1D de la séquence (p_1, p_2).

Le point de référence du prochain quintuplet est le point p_2 .

Longueur	Séquence blanche	Séquence noire
	code pour R	
0	`00110101`	`0000110111`
1	`000111`	`010`
2	`0111`	`11`
⋮	⋮	⋮
62	`00110011`	`000001100110`
63	`00110100`	`000001100111`
	code pour $(64 * Q)$	
64 (1)	`11011`	`0000001111`
128 (2)	`10010`	`000011001000`
⋮	⋮	⋮
1664 (26)	`011000`	`0000001100100`
1728 (27)	`010011011`	`0000001100101`
`EOL`	`000000000001`	`000000000001`

Tableau 3.1. *Table de Huffman modifiée du CCITT G3*

Ces modes sont identifiés grâce à un quintuplet de pixels $< p_0, p_1, p_2, q_0, q_1 >$:

– p_0 est situé sur la ligne en cours de codage. Sa position est déterminée par le précédent mode. Au début du codage de la ligne, il est positionné de manière fictive devant tous les pixels de la ligne ;

– p_1 est le premier pixel après p_0 (à droite) sur la même ligne qui soit de couleur opposée à celle de p_0 ;

– p_2 est le pixel suivant (à droite) sur la même ligne de même couleur que p_0 ;

– q_0 est le premier pixel de la ligne de référence situé à droite de p_0 et ayant une couleur opposée à ce dernier ;

– q_1 est le pixel à droite de p_0 sur la ligne de référence et ayant la même couleur que p_0; donc de couleur opposée à celle de q_0.

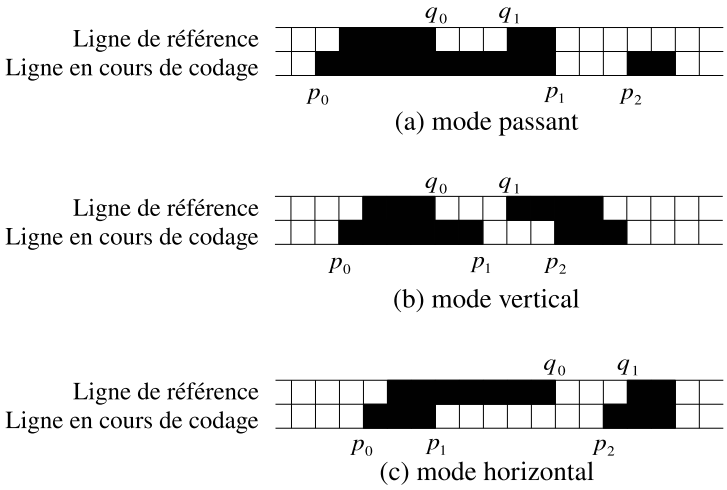


Figure 3.22. *Codage RLC-2D*

Ces techniques de codage par plages ne sont intéressantes qu'en utilisation mixte avec une autre méthode de codage. On le constate avec les standards du CCITT où un codage de Huffman modifié est utilisé. Ainsi, le codage par plages est utilisé par les différents formats de compression d'images et de vidéos décrits dans les prochains chapitres. Enfin, on retrouve le codeur par plages, dans beaucoup de compresseurs actuels, associé à d'autres techniques plus élaborées, mais, plus coûteuses. Par exemple, le codeur JPEG-LS l'utilise. Lorsque celui-ci détecte une plage de valeurs identiques, il fait appel au codeur par plage plutôt qu'au codeur de Golomb (voir paragraphe 3.4.3.2 page 113) de son mode normal. Des versions simplifiées de codeur par plages sont définis par les standards de compression d'images et de vidéos.

3.4.2. *Codeurs à dictionnaires dynamiques*

Pour des données provenant d'un alphabet plus riche, les techniques de codage par plages sont vite limitées. En effet, de part la richesse même des alphabets des sources usuellement codées, les séquences de valeurs identiques sont plus rares. En revanche, on peut observer des périodes de valeurs, parfois appelées *motifs* (ou *patterns* par anglicisme). Par exemple, si le signal à coder est une suite de caractères avec le code ASCII pour alphabet, chaque caractère est alors codé sur 8 bits. Plusieurs caractères consécutifs forment un motif susceptible d'apparaître plusieurs fois dans le texte. Si c'est le cas, on peut associer un code à cette séquence et remplacer l'ensemble de ses occurrences par ce code. Il suffit alors d'étendre le codage : de 8 à 12 bits par exemple.

C'est ce principe qui est repris par les codeurs à dictionnaires dynamiques. Ils utilisent tous un dictionnaire initial qui augmente à chaque apparition de nouvelles périodes. Parmi tous les codeurs existants, cette section décrit le plus connus d'entre eux : le codeur LZW des initiales de ses concepteurs Lempel, Ziv et Welch. L'algorithme du codeur, appliqué aux valeurs ASCII, est le suivant :

```
CODEUR_LZW()
1  S ←'' // code vide
2  tant que la source n'est pas vide
3  faire LIRE la valeur c dans le flux d'entrée
4        si (S + c) existe déjà dans le dictionnaire
5           alors S ← S + c // concaténer les bits de c en fin de S
6           sinon EMETTRE code (S) dans le flux de sortie
7                 AJOUTER(S + c) dans le dictionnaire
8                 S ← c // réinitialisation de la chaîne
9  EMETTRE code (S) dans le flux de sortie
```

EXEMPLE 3.6.– *Soit la source à coder : « Bacchanales Bacchus ». L'alphabet initial est la table ASCII codée sur 12 bits afin de pouvoir l'étendre. En suivant pas à pas le codeur, on obtient le tableau 3.2; le premier mot de la source sert à construire les nouveaux codes (256 à 266) du dictionnaire qui sont ensuite utilisés pour le codage du deuxième mot.*

REMARQUE 3.7.–

1) les codes sont envoyés alors que toute la source n'a pas été décodée. Le décodeur peut donc commencer son traitement sans attendre la fin du codage;

2) le dictionnaire n'est pas envoyé au récepteur. Le décodeur reconstruit à l'aveugle ce dictionnaire. C'est l'avantage des codeurs/décodeurs à dictionnaires dynamiques.

entrée	valeurs émises	nouveau code	chaîne S après modif.	commentaires
			”	
B			’B’	’B’ $\in$ dictionnaire
a	ascii(B)=66	256=code(’Ba’)	’a’	’BA’$\notin$ dictionnaire
c	ascii(a)=97	257=code(’ac’)	’c’	’ac’$\notin$ dictionnaire
c	ascii(c)=99	258=code(’cc’)	’c’	’cc’$\notin$ dictionnaire
h	ascii(c)=99	259=code(’ch’)	’h’	’ch’$\notin$ dictionnaire
a	ascii(h)=104	260=code(’ha’)	’a’	’ha’$\notin$ dictionnaire
n	ascii(a)=97	261=code(’an’)	’n’	’an’$\notin$ dictionnaire
a	ascii(n)=110	262=code(’na’)	’a’	’na’$\notin$ dictionnaire
l	ascii(a)=97	263=code(’al’)	’l’	’al’$\notin$ dictionnaire
e	ascii(l)=108	264=code(’le’)	’e’	’le’$\notin$ dictionnaire
s	ascii(e)=101	265=code(’es’)	’s’	’es’$\notin$ dictionnaire
	ascii(s)=115	266=code(’s ’)	’ ’	’s ’$\notin$ dictionnaire
B	ascii( )=32	267=code(’ B’)	’B’	’ B’$\notin$ dictionnaire
a			’Ba’	’Ba’$\in$dictionnaire
c	256	268=code(’Bac’)	’c’	’Bac’$\notin$ dictionnaire
c			’cc’	’cc’$\in$dictionnaire
h	258	269= code(’cch’)	’h’	’cch’$\notin$ dictionnaire
u	ascii(h)=104	270=code(’hu’)	’u’	’hu’$\notin$ dictionnaire
s	ascii(u)=117	271=code(’us’)	’s’	’us’$\notin$ dictionnaire
	ascii(s)=115			

Tableau 3.2. *Simulation du codage de la source « Bacchanales Bacchus » : la première colonne informe sur la lecture courante dans le flux d'entrée; la deuxième colonne sur l'émisssion dans le flux de sortie; la troisième colonne indique la nouvelle valeur enregistrée dans le dictionnaire; la quatrième colonne informe sur la valeur de la chaîne de caractères S après sa mise à jour (concaténation ou réinitialisation).*

L'algorithme du décodeur est le suivant :

```
DÉCODEUR_LZW()
 1  S ←''
 2  tant que la source n'est pas vide
 3  faire LIRE le code c
 4       si c est un index du dictionnaire initial
 5          alors m ← VALEUR(c)
 6                ECRIRE m
 7                si (S + m) existe déjà dans le dictionnaire
 8                   alors S ← S + m // concaténer c en fin de S
 9                   sinon AJOUTER(S + m) dans le dictionnaire
10                         S ← c
11          sinon METTRE m en tête de lecture
```

Le décodage est effectué uniquement avec les codes du dictionnaire intial (la table ASCII). Les codes supplémentaires sont *reconstruits* et servent, d'une certaine manière, d'indexage double.

L'exemple de décodage qui suit prend en donnée le code généré par le codeur de l'exemple précédent.

EXEMPLE 3.7.– *Le décodeur initialise son dictionnaire avec la table ascii. Puis, il lit les codes envoyés par le codeur :*

*66,97,99,99,104,97,110,97,108,101,115,32,**256**, **258**, 104,117,115*

Tant que les codes sont inférieurs à `256`, ils appartiennent au dictionnaire initial et peuvent donc être directement décodés; au code `66` correspond le caractère B qui est affiché. En même temps, les codes supplémentaires sont construits comme le fait le codeur.
De cette manière, quand le code est supérieur ou égal à `256`, le décodeur est en présence d'un code supplémentaire qu'il a reconstruit auparavant. Il peut donc remplacer ce code par sa séquence de codes provenant du dictionnaire initial. Le résultat est détaillé dans le tableau 3.3.

Ce type de codage convient parfaitement à des données de type texte. On le retrouve dans des outils systèmes de compression tels que les formats de suffixes ".Z", ".zip" ou encore ".gzip". Il est également utilisé par le format GIF spécialisé dans la compression d'images de synthèse. La variante LZ77 est utilisée par la compression sans perte de type déflation[16] de textes, de données système[17] et d'images au format PNG (voir section PNG, du chapitre Compression sans perte du volume 2). LZ77 effectue la recherche des occurrences à l'aide d'une fenêtre glissant sur les lignes de caractères. Cette fenêtre est scindée en deux : une zone de recherche et une zone de prélecture. La zone de recherche conserve les derniers caractères codés afin d'y retrouver la plus grande période présente en zone de prélecture.

3.4.3. *Codeurs à longueurs variables*

Un codeur à longueurs variables (VLC). affecte une séquence de bits de longueur l_i différente pour chaque donnée x_i (voir paragraphe 3.3.1.3). La source, $S = \{x_0, x_1, \ldots, x_n\}$, est codée sur une séquence de bits de longueur :

$$L_S = \sum_{x_i \in S} l_i$$

16. Traduction libre pour *deflate*.
17. Formats Zip et Gzip par exemple.

De cette manière, il est possible pour le débit binaire $R = L_S/m$, de mieux approcher la valeur de l'entropie, tout en s'assurant de ne pas la dépasser, quelle que soit la source à coder.

Le principe repose sur la propriété de *codes préfixés* qui a comme propriété qu'aucun code ne peut être le début d'un autre. Ainsi, lors du décodage, il n'y a pas d'ambiguïté sur les codes à retrouver, bien que ne connaissant pas leurs longueurs.

EXEMPLE 3.8.– *Soit un alphabet constitué de quatre symboles : $A = \{0, 1, 2, 3\}$ Pour les coder avec un minimum de bits, on peut assigner un seul bit pour les deux premiers symboles et deux bits pour les deux suivants :*

$$\begin{aligned} c_0 &= 0 \\ c_1 &= 1 \\ c_2 &= 10 \\ c_3 &= 11 \end{aligned}$$

lecture	affichage	code supplémentaire	la chaîne S après modif	commentaires
			”	
66	B		’B’	
97	a	256=code(’Ba’)	’a’	
99	c	257=code(’ac’)	’c’	
99	c	258=code(’cc’)	’c’	
104	h	259=code(’ch’)	’h’	
97	a	260=code(’ha’)	’a’	
110	n	261=code(’an’)	’n’	
97	a	262=code(’na’)	’a’	
108	l	263=code(’al’)	’l’	
101	e	264=code(’le’)	’e’	
115	s	265=code(’es’)	’s’	
32	espace	266=code(’s ’)	’ ’	
256	———	———————	——	*subtitution de 256*
	———	———————	——	*par (66,97)*
66	*B*	*267=code(’ B’)*	*’B’*	
97	*a*		*’Ba’*	
258	———	———————	——	*substitution de 258*
	———	———————	——	*par (99,99)*
99	*c*	*268=code(’Bac’)*	*’c’*	
99	*c*		*’cc’*	
104	h	269=code(’cch’)	’h’	
117	u	270=code(’hu’)	’u’	
115	s	271=code(’us’)	’s’	

Tableau 3.3. *Simulation de décodage*

Si le codeur reçoit la source $S = 10\,233$*, il enverra la séquence* 10101111*. Plusieurs interprétations sont alors possibles pour le décodeur. Il peut, par exemple, comprendre* $\hat{S} = 10101111$ *ou bien* $\hat{S} = 2\,233$ *mais aussi* $\hat{S} = 10\,233$*. Ceci est dû au fait que* c_1 *est le préfixe de* c_2 *et de* c_3 *: son code est le début des deux autres.*
Pour éviter cette ambiguïté, on choisit une construction automatique des codes. Par exemple, le symbole c_i *est représenté par une séquence de* i *bits à* 0 *se terminant par un bit à* 1 *:*

$$\begin{aligned} c_0 &= 1 \\ c_1 &= 01 \\ c_2 &= 001 \\ c_3 &= 0001 \end{aligned}$$

Ce *codage préfixé* est un classique. Les bits à 1 servent de délimiteurs entre les séquences de bits à 0. Aussi, est il appelé *code unaire*.

Le codage préfixé assure un décodage *instantané*, mais pas forcément avec un débit binaire L/m, plus proche de l'entropie que dans le cas d'un codage à longueur fixe.

Dans l'exemple précédent, un codage à longueur fixe consisterait à assigner deux bits à chacun des symboles : $L = 2$ et $m = 1$. La source $S = 10\,233$ est alors codée sur 10 bits. En revanche, avec le code *unaire*, la source est codée sur 14 bits !

Cependant, si la distribution est de la forme :

$$P(X = x_i) = 2^{-(i+1)} \forall x_i \in A \tag{3.4}$$

le débit moyen du code unaire pour une source S, sans mémoire[18] vaut :

$$\begin{aligned} R &= \sum_{x_i \in S} \underbrace{(i+1)}_{\text{nombre de bits pour coder } x_i} P(X = x_i) \\ &= \sum_{x_i \in S} (i+1) 2^{-(i+1)} \\ &= \sum_{x_i \in S} -\log_2(P(X = x_i)) P(X = x_i) \\ &= H(S) \end{aligned} \tag{3.5}$$

18. Les possibles relations entre les valeurs au sein de la source sont ignorées.

Ainsi, le débit binaire atteint la valeur de l'entropie de la source et le codage à l'aide du code unaire est optimal (voir théorème 3.5).

Le théorème fondamental, qui suit, étend ce résultat à d'autres distributions.

THÉORÈME 3.6.– *Pour toute distribution de probabilité $P(X = x_i)$, il existe un code préfixé tel que :*

$$H(X) \leqslant R < H(X) + 1$$

3.4.3.1. *Codage de Huffman*

Huffman fut l'un des premiers a proposé un algorithme de construction de code préfixé qui respecte le théorème 3.6. L'idée est d'attribuer peu de bits aux symboles de forte probabilité, c'est-à-dire apparaissant souvent.

On suppose l'alphabet $A = \{x_0, x_1, \cdots, x_{n-1}\}$ tel que sa distribution vérifie :

$$p(X = x_0) \leqslant p(X = x_1) \leqslant \cdots \leqslant p(X = x_{n-1})$$

Pour un codage optimal, les longueurs des codes doivent alors respecter :

$$l_0 \geqslant l_1 \geqslant \cdots \geqslant l_{n-1}$$

Les deux symboles de faibles probabilités x_0 et x_1, sont obligatoirement de longueurs identiques :

$$l_0 = l_1$$

En effet, si l'on suppose que $l_0 \neq l_1$, on a alors forcément $l_1 < l_0$. Mais, comme le code de x_1 ne doit pas être un préfixe du code de x_0, on peut les distinguer grâce à leurs l_1 bits de poids fort :

$$b_0^{x_0} b_1^{x_0} \cdots b_{l_1-1}^{x_0} \neq b_0^{x_1} b_1^{x_1} \cdots b_{l_1-1}^{x_1}$$

Ainsi, les $(l_0 - l_1)$ bits de poids faible de x_0 sont inutiles. On peut donc assigner à ces deux symboles des codes identiques, sauf en leur dernier bit.

Ensuite, en regroupant ces deux symboles en un seul, nous construisons un nouvel alphabet :

$$A' = \{x'_1, x_2, \cdots, c_{n-1}\}$$

Les probabilités sont mises à jour :

$$p(X' = x) = \begin{cases} p(X = x_o) + p(X = x_1) & \text{si } x = x'_1 \\ p(X = x_i) & \text{sinon} \end{cases}$$

En réordonnant cet alphabet suivant l'ordre croissant des probabilités, on retrouve l'état initial, mais, avec un alphabet possédant un symbole de moins.

Huffman propose un algorithme qui suit ce schéma de manière récursive jusqu'à obtenir un alphabet constitué d'un unique symbole. A chaque regroupement, il faut mémoriser les deux symboles concernés et leurs derniers bits respectifs (ceux qui permettent de les distinguer).

EXEMPLE 3.9.– *Soit l'alphabet* $\{a, c, d, e, f, g, h, l, m, n, o, p, u, x, \perp\}$ *et le message « exemple* $\perp$ *de* $\perp$ *codage* $\perp$ *de* $\perp$ *huffman » à coder. La distribution de probabilité est estimée directement à partir des fréquences d'apparition des symboles dans la source :*

$$\begin{array}{lcllcl} P(X=a) & = & 2/28 & P(X=c) & = & 1/28 \\ P(X=d) & = & 3/28 & P(X=e) & = & 6/28 \\ P(X=f) & = & 2/28 & P(X=g) & = & 1/28 \\ P(X=h) & = & 1/28 & P(X=l) & = & 1/28 \\ P(X=m) & = & 2/28 & P(X=n) & = & 1/28 \\ P(X=o) & = & 1/28 & P(X=p) & = & 1/28 \\ P(X=u) & = & 1/28 & P(X=x) & = & 1/28 \\ P(X=\perp) & = & 4/28 & & & \end{array}$$

On réordonne l'alphabet suivant l'ordre croissant des probabilités :

$$\{c, g, h, l, n, o, p, u, x, a, f, m, d, \perp, e\}$$

Puis, les deux premiers symboles, c *et* g*, sont regroupés en un nouveau,* z*. Ensuite, les symboles* c *et* g*, sont mémorisés avec leur dernier bit à* 0 *et à* 1 *respectivement. On obtient le nouvel alphabet :*

$$\{h, l, n, o, p, u, x, a, f, m, z, d, \perp, e\}$$

avec les probabilités :

$$\frac{1}{28}\{1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 4, 6\}$$

Puis, on réitére le procédé avec ce nouvel alphabet.
La figure 3.23 illustre l'ensemble du processus sous sa forme arborescente. La construction se fait de gauche à droite. Chaque nœud interne représente le regroupement des deux symboles les moins probables. Leurs derniers bits respectifs sont

indiqués sur les arcs du regroupement. Une fois l'arbre construit, les codes se lisent dans un parcours en sens inverse, de la racine vers les feuilles. On obtient les codes suivants :

$$
\begin{array}{rl}
e \rightarrow 11 & \perp \rightarrow 110 \\
d \rightarrow 101 & m \rightarrow 1010 \\
f \rightarrow 0010 & a \rightarrow 1001 \\
x \rightarrow 0001 & u \rightarrow 11100 \\
p \rightarrow 01100 & o \rightarrow 10100 \\
n \rightarrow 00100 & l \rightarrow 11000 \\
h \rightarrow 01000 & g \rightarrow 10000 \\
c \rightarrow 00000 &
\end{array}
$$

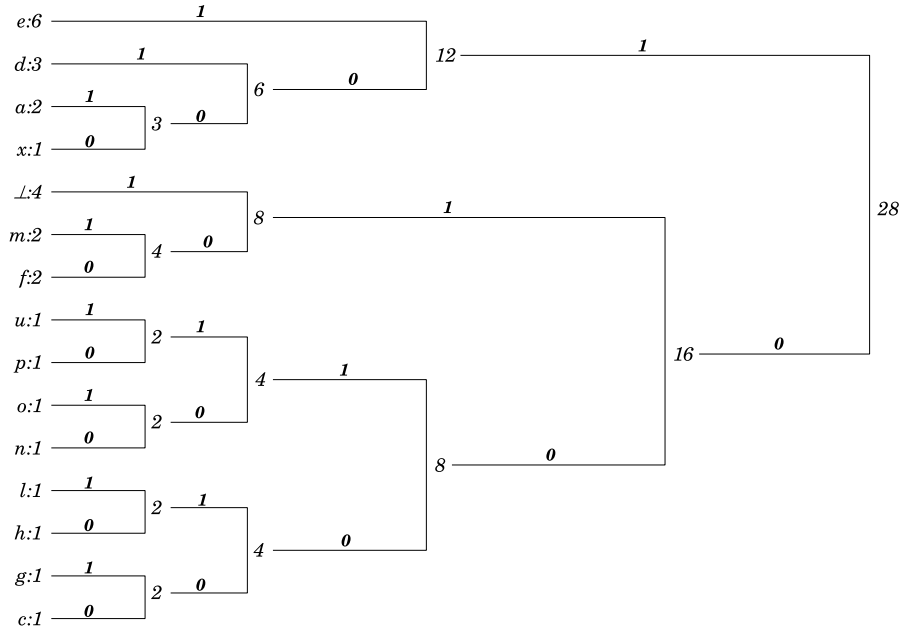


Figure 3.23. *Codage de Huffman pour le message « $exemple \perp de \perp codage \perp huffman$ »*

Cet algorithme, parmi les pionniers dans la construction automatique de codes entropiques, comporte quelques inconvénients. Certains – comme le fait que les distributions doivent être connues *a priori* ou bien estimées à partir de jeux d'essais – sont communs a bon nombre d'algorithmes. D'autres sont une limite certaine de l'algorithme tel qu'il est présenté. Par exemple, le codage obtenu ne peut, par construction même, avoir un débit binaire en deçà d'un bit par symbole, alors que l'entropie peut l'être naturellement. Il reste alors des redondances dans le code généré.

Pour éliminer ces redondances, une version de l'algorithme propose de construire un nouvel alphabet, dont chaque symbole est le regroupement de plusieurs symboles de l'alphabet initial. Le débit binaire minimum possible reste toujours d'un bit par symbole. Mais, comme ce nouveau symbole regroupe n symboles de l'alphabet initial, la borne inférieure du débit binaire pour l'alphabet initial est de $1/n$ bits par symbole. Les nouveaux symboles sont des *blocs* de symboles et le codage qui en découle fournit des *code-blocs*.

La construction de la table de codage peut également être une limite de l'algorithme. Si la table des codes est construite *dynamiquement* à partir des données de la source, elle doit alors être transmise au récepteur, ce qui augmente le débit. Il est possible d'alléger la transmission entre l'émetteur et le récepteur en utilisant une version *statique* de la table, qui est obligatoirement stipulée dans les algorithmes du codeur et du décodeur. Ceci signifie qu'elle est construite en amont par les constructeurs/développeurs et donc adaptée à certaines sources plus qu'à d'autres.

Cet algorithme est utilisé pour son efficacité et sa simplicité de mise en œuvre, en particulier, par les formats JPEG et MPEG. Toutefois, les codeurs arithmétiques (voir paragraphe 3.4.4), développés plus récemment, offrent une alternative attrayante bien que plus complexe à mettre en œuvre.

3.4.3.2. *Codage de Golomb*

Son principe reprend le cas d'une *distribution géométrique* (voir equation (3.4)) pour laquelle le *code unaire* est optimal. La généralisation de cette distribution considère que les probabilités p et $q = (1 - p)$ prennent des valeurs quelconques :

$$P(X = x) = qp^x = (1 - p)p^x$$

Son écriture s'interprète comme la probabilité d'atteindre l'état défini par la probabilité q au bout de x itérations du processus. Avec $p = \frac{1}{2}$, on retrouve l'écriture de l'équation (3.4).

EXEMPLE 3.10.– *Dans le jeu du pile ou face (de probabilités q pour pile et p pour face), si la pièce n'est pas truquée, $p = q = \frac{1}{2}$ et la probabilité d'obtenir pile au $(x + 1)^e$ lancement de la pièce vaut 2^{-x}.*

Mais, la pièce peut être truquée de telle manière que le côté pile apparaisse plus souvent que le côté face, $q > p$... Par exemple, $p = \frac{1}{3}$. Dans ce cas, la probabilité d'avoir pile au 5^e lancement vaut $\frac{2}{3}.\left(\frac{1}{3}\right)^4$.

Comme l'illustre la figure 3.24, cette probabilité va en décroissant avec le nombre de lancers, mais commence avec une plus forte probabilité pour le premier lancer de la pièce.

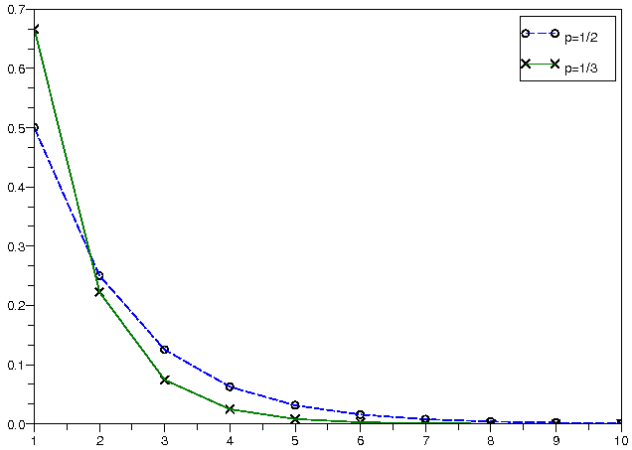


Figure 3.24. *Distribution géométrique pour le jeu du pile ou face avec une pièce non truquée* $\left(p = \frac{1}{2}\right)$ *et une pièce truquée* $\left(p = \frac{1}{3}\right)$

Le code unaire est optimal pour des valeurs $p \leqslant \frac{1}{2}$ [TAU 02] et inefficace pour des valeurs de $p > \frac{1}{2}$. Mais, en écrivant x sous la forme d'un quotient et d'un reste :

$$x = mx_d + x_r$$

La distribution associée au quotient, x_d, est de la forme :

$$\begin{array}{rcl} P(X_d = x_d) & = & \sum_{i=0}^{m-1} P(X = mx_d + i) \\ & = & \sum_{i=0}^{m-1} P(X = mx_d)P(X = i) \\ & = & (1-p)p^{mx_d} \sum_{i=0}^{m-1} P(X = i) \\ & \approx & (1-p^m)p^{mx_d} \sum_{i=0}^{m-1} P(X = i) \end{array}$$

L'algorithme de Golomb va rechercher la valeur m telle que p^m soit supérieurement proche de $\frac{1}{2}$ $\left(p^m \gtrsim \frac{1}{2}\right)$ car, dans ce cas, le codage unaire devient quasi optimal pour le codage de x_d (voir equation 3.5 en page 109).

De plus, la distribution du reste, x_r, peut être supposée uniforme car les valeurs de probabilité fluctuent dans l'intervalle restreint $\left[(1-p), \frac{1}{2}(1-p)\right]$:

$$(1-p) = P(X = 0) \geqslant \cdots \geqslant P(X = m-1) > P(X = m) = (1-p)p^m \gtrsim \frac{1}{2}(1-p)$$

Le codage binaire est, lui aussi, quasi optimal. En pratique, il est utile de considérer m comme une puissance de 2 :

$$m = 2^k$$

EXEMPLE 3.11.– *Pour le paramètre $k = 3$, la valeur $x = 37$ est décomposable en un quotient et un reste :*

$$\begin{cases} x_d = \lfloor x/2^k \rfloor = 4 \\ \\ x_r = x \bmod 2^k = 5 \end{cases}$$

x_d est codé en unaire tandis que le reste x_r est codé en binaire à l'aide des k bits de poids faible de x :

$$(37)_{10} = (100101)_2 = (\underbrace{00001}_{x_d} | \underbrace{101}_{x_r})_{\text{Golomb}_{k=3}}$$

S. W. Golomb montre que m peut être approximé comme étant la moyenne statistique, $E(X)$, de la source [SAL 07, TAU 02] :

$$m = 2^k \gtrsim \tfrac{E(X)}{2} \iff k = \max\left(0, \left\lceil \log_2\left(\tfrac{E(X)}{2}\right)\right\rceil\right)$$

L'algorithme est alors le suivant :

ALGORITHME DE GOLOMB()

```
1  // μ̂_X est l'estimation initiale de l'espérance E(X)
2  // A/N est l'estimation de l'espérance E(X)
3  A ← μ̂_X
4  N ← 1
5  pour i ← 0 à n
6  faire k ← max(0, ⌈log2(A/(2N))⌉)
7        coder x_i avec le paramètre k
8        si N = N_max
9          alors // mise à jour du compteur N et du cumul A
10               A ← ⌊A/2⌋
11               N ← ⌊N/2⌋ A ← A + x_i
12       N ← N + 1
```

$N_{\max}$ est une constante de programmation pour ne pas dépasser la taille allouée au codage. Le décodeur opère les mêmes mises à jour et estime également le paramètre k. Munie de ce paramètre, la valeur x_i peut être reconstruite. Cet algorithme est utilisé par le standard JPEG-LS lors de la compression sans perte d'images (voir section JPEG-LS, chapitre « Compression sans perte » du volume 2).

3.4.4. *Codage arithmétique*

Peu de temps après la publication par Claude Shannon du théorème 3.5, P. Elias propose un algorithme qui, bien que purement théorique, s'avère être l'initiateur de la famille des codeurs dits arithmétiques.

Dans le cas d'un processus aléatoire sans mémoire, chaque instant t est décrit par la même VA X d'alphabet A et de distribution de probabilité $\{p_i = P(X = x_i), \forall x_i \in A\}$.

A l'instant $t = 0$, la VA X vaut x_0 avec la probabilité p_0. A l'instant suivant $(t = 1)$, la VA X vaut x_1 avec la probabilité p_1 et ainsi de suite.

Partant de la définition des probabilités :

$$0 \leqslant p_i \leqslant 1 \text{ et } \sum_{i=0}^{m-1} p_i = 1$$

L'algorithme de P. Elias fixe une valeur réelle dans l'intervalle $[0, 1[$ qui correspond à la probabilité du processus aléatoire. Dans le cas où la source est sans mémoire (avec des VA indépendantes), la probabilité du processus est le produit des probabilités de l'ensemble des événements réalisés :

$$P(X_0 = x_0, X_1 = x_1, \cdots, X_{n-1} = x_{n-1}) = \prod_{i=0}^{n-1} p_i$$

L'intervalle initial $[0, 1[$ sera noté $[b, b + t[$ avec $b_0 = 0$ et $t_0 = 1$. Il va être affiné à chaque nouvel événement apparu à l'aide des probabilités p_i assignées aux événements.

A l'initialisation, l'intervalle $[0, 1[$ est divisé suivant les valeurs des probabilités p_i : le premier sous-intervalle est $[0, p_0[$, le second $[p_0, (p_0 + p_1)[$ et donc le nième $[b_{n-1}, b_{n-1} + t_{n-1}]$ avec $b_{n-1} = \sum_{i=0}^{n-2} p_i$ et $t_{n-1} = p_{n-1}$.

Si l'on suppose, qu'à l'instant $t = 0$, la VA X vaut x_u. L'intervalle $[0, 1[$ est réduit à l'intervalle $[b_u, b_u + t_u[$ avec :

$$\begin{cases} b_1 = b_0 + t_0 * \sum_{j=0}^{u-1} p_j \\ t_1 = t_0 * p_u \end{cases}$$

Ensuite, à l'instant $t = 1$, la réalisation x_v vient réduire l'intervalle $[b_u, b_u + t_u[$. Le nouvel intervalle est de la forme $[b_v, b_v + t_v[$ avec :

$$\begin{cases} b_2 = b_1 + t_1 * \sum_{j=0}^{v-1} p_j \\ t_2 = t_1 * p_v \end{cases}$$

A $t = 2$, la réalisation x_w vient, à son tour, réduire l'intervalle $[b_v, b_v + t_v[$ avec :

$$\begin{cases} b_3 = b_2 + t_2 * \sum_{j=0}^{w-1} p_j \\ t_3 = t_2 * p_w \end{cases}$$

On voit apparaître un schéma récursif de réduction de l'intervalle initial $[0, 1[$. Cette réduction n'utilise que les probabilités p_i et les fonctions cumulatives $F(x_i) = \sum_{j=0}^{i-1} p_j$:

CODAGE DE P. ELIAS()

1 *// initialisation de la borne et de la taille*
2 $b_0 \leftarrow 0\,;\ t_0 \leftarrow 1$
3 *// pour chaque événement*
4 **pour** $i \leftarrow 0$ **à** $n - 1$
5 **faire** $b_{i+1} \leftarrow b_i + t_i F(x_i)\,;\ t_{i+1} \leftarrow t_i p_i$
6 **retourner** α tel que $\alpha \in [b_n, b_n + t_n[$

Cet algorithme possède les propriétés suivantes :

– $[b_{i+1}, b_{i+1} + t_{i+1}[\subset [b_i, b_i + t_i[$;

– $t_n = \prod_{i=0}^{n-1} p_i = P(X_0 = x_0, X_1 = x_1, \cdots, X_{n-1} = x_{n-1})$ pour des VA indépendantes.

Il fournit l'intervalle $[b_n, b_n + t_n[$ dans lequel n'importe quelle valeur α peut être choisie pour représenter les n événements; en particulier $\alpha = b_n$. A partir de la valeur α, le décodeur identifie les intervalles des n événements x_i. Pour ce faire, il doit connaître l'alphabet A et les probabilités p_i. Le principe du décodeur est de suivre le codeur, pas à pas, dans ses étapes d'affinage de l'intervalle :

DÉCODAGE DE P. ELIAS()

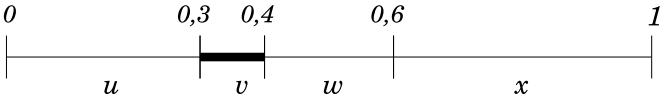
1 $b_0 \leftarrow 0\,;\ t_0 \leftarrow 1$
2 **pour** $i \leftarrow 0$ **à** $n - 1$
3 **faire** $k \leftarrow 0$
4 **répéter**
5 $b_{i+1} \leftarrow b_i + t_i \sum_{j=0}^{k-1} p_j\,;\ t_{i+1} \leftarrow t_i p_k$
6 $k \leftarrow k + 1$
7 **jusqu'à** $b_{i+1} \leqslant \alpha < b_i + t_i$
8 EMETTREx_{i+1}

REMARQUE 3.8.– *Il faut définir un code spécifique pour que le décodeur sache quand terminer.*

EXEMPLE 3.12.– *Soit l'alphabet* $A = \{u, v, w, x\}$ *de distribution de probabilité* $\{0.3, 0.1, 0.2, 0.4\}$. *La source à coder est « vwu ». La simulation de l'algorithme de codage donne :*

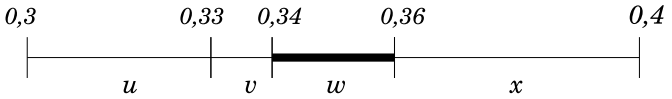
1) initialisation de l'intervalle : $b_0 = 0$ *et* $t_0 = 1$ *;*

2) lecture de la première donnée : v *de probabilité* $0,1$ *:*



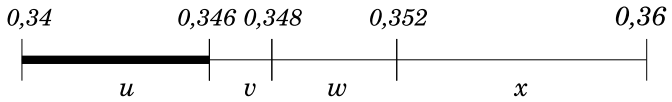
Réduction de l'intervalle : $b_1 = b_0 + t_0 * (p_0) = 0,3$ *et* $t_1 = t_0 * p_1 = 0,1$ *;*

3) lecture de la deuxième donnée : w *de probabilité* $0,2$ *:*



Réduction de l'intervalle : $b_2 = b_1 + t_1 * (p_0 + p_1) = 0,34$ *et* $t_2 = t_1 * p_2 = 0,02$ *;*

4) lecture de la troisième donnée : u *de probabilité* $0,3$ *:*



Réduction de l'intervalle : $b_3 = b_2 + t_2 * (0) = 0,34$ *et* $t_3 = t_2 * p_0 = 0,006$ *;*

5) envoi de la valeur $\alpha = b_3 = 0,34$

Le décodeur reçoit la valeur α *qu'il utilise pour retrouver les intervalles et émettre les valeurs correspondantes :*

1) initialisation de l'intervalle : $b_0 = 0$ *et* $t_0 = 1$ *;*

2) recherche du sous-intervalle :

a) $b_1 = b_0 + t_0 * (0) = 0$ *et* $t_1 = t_0 * p_0 = 0,3 \Rightarrow$ *mauvais intervalle :* $b_1 + t_1 \leqslant \alpha$ *,*

b) $b_1 = b_0 + t_0 * (p_0) = 0,3$ *et* $t_1 = t_0 * p_1 = 0,1 \Rightarrow$ *bon intervalle :* $b_1 \leqslant \alpha < b_1 + t_1$ *.*

Emission du code v *et l'intervalle vaut* $[0,3 \quad 0,4[$ *;*

3) Recherche du sous-intervalle :

a) $b_2 = b_1 + t_1 * (0) = 0,3$ *et* $t_2 = t_1 * p_0 = 0,03 \Rightarrow b_2 + t_2 \leqslant \alpha$ *,*

b) $b_2 = b_1 + t_1 * (p_0) = 0,33$ *et* $t_2 = t_1 * p_1 = 0,01 \Rightarrow b_2 + t_2 \leqslant \alpha$ *,*

c) $b_2 = b_1 + t_1 * (p_0 + p_1) = 0,34$ *et* $t_2 = t_1 * p_2 = 0,02 \Rightarrow b_2 \leqslant \alpha < b_2 + t_2$ *.*

Emission du code w *et l'intervalle vaut* $[0,34 \quad 0,36[$ *;*

4) recherche du sous-intervalle :

a) $b_2 = b_1 + t_1 * (0) = 0,34$ *et* $t_3 = t_2 * p_0 = 0,006 \Rightarrow b_3 \leqslant \alpha < b_3 + t_3$

Emission du code u *et l'intervalle vaut* $[0,34 \quad 0,346[$ *.*

Cette technique de codage a rencontré plusieurs inconvénients qui l'ont empéchée d'être appliquable pendant longtemps. En effet, plus la source à coder est importante en taille, plus les valeurs de la borne et de la taille de l'intervalle augmentent en précision, alors que la représentation machine des réels est limitée en précision !

Mais, le codage arithmétique a été amélioré successivement par l'introduction du codage arithmétique à précision finie, décrit en annexe B.6, puis par l'introduction du codage arithmétique binaire, détaillé en annexe B.7. Ce dernier est de plus en plus utilisé. Il est souvent associé à l'établissement de *contextes* qui modélisent le voisinage des échantillons à coder.

3.5. Prédictions

Parmi les propriétés caractérisant les échantillons à coder, l'interdépendance (ou corrélation) est significative. Elle indique qu'une donnée à un instant précis ou/et un lieu précis[19] dépend fortement des données qui la précèdent. Cette dépendance se traduit par une différence relativement faible entre la donnée courante et ses prédécesseurs. Le codage de cette différence est alors forcément moins coûteux que le codage de la donnée proprement dite.

Si l'on considère une source, de VA X, dont les amplitudes suivent une progression linéaire avec le temps (voir figure 3.25a), il est alors évident que le codage de la différence entre une donnée et son prédécesseur est nettement moins coûteux que le codage des données elles-mêmes. Dans l'exemple de la figure 3.25a, les amplitudes suivent la loi :

$$x_t = x_{t-1} + 1 \text{ avec } x_0 = 2 \text{ et } t \in \mathbb{N}^+$$

La différence, dont la VA est notée D, est donc constante et de valeur 1. Son codage est relativement simple et un codeur par plages, légèrement modifié, est déjà très efficace : le décodeur a besoin de connaître la première donnée (de valeur 2), la différence (de valeur 1) et le nombre de données pour reconstruire le signal à l'identique.

Cette idée se comprend également à la lecture de la mesure d'entropie. Si l'entropie de la différence est nettement plus faible que celle de la source alors il faut coder cette différence, plutôt que la source [PLU 94].

---

19. *A un instant précis*, s'il s'agit d'une donnée temporelle, ou bien, *à une position précise* s'il s'agit d'une donnée spatiale. Par défaut, on parle de données temporelles, sachant que la généralisation aux données spatiales et spatio-temporelles est possible.

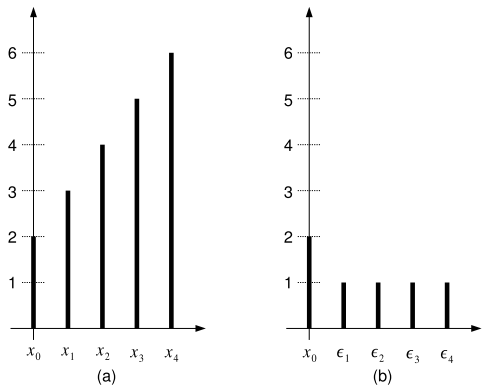


Figure 3.25. *(a) les données x_t et (b) les différences $\epsilon_t = x_t - x_{t-1}$*

Dans l'exemple précédent, l'entropie de D est évidemment très faible puisque la valeur 1 a une probabilité très élevée :

$$\begin{aligned} H(D) &= -\sum_{x_i \in \{x_0,1\}} p_{x_i} \log_2(p_{x_i}) \\ &= -P(X = x_0) \log_2(P(X = x_0)) - P(X = 1) \log_2(P(X = 1)) \\ &= -\left(\tfrac{1}{n}\right) \log_2 \left(\tfrac{1}{n}\right) - \left(\tfrac{n-1}{n}\right) \log_2 \left(\tfrac{n-1}{n}\right) \\ &= \log_2(n) + \tfrac{1-n}{n} \log_2(n-1) \end{aligned}$$

Alors que l'entropie de la source vaut :

$$\begin{aligned} H(X) &= -\sum_i p_i \log_2(p_i) \\ &= -n\tfrac{1}{n} \log_2(\tfrac{1}{n}) \\ &= \log_2(n) \end{aligned}$$

La figure 3.26 montre les graphes de ces deux entropies en fonction du nombre d'échantillons n, de la source. On constate que l'entropie va en croissant pour la source, alors qu'elle diminue régulièrement pour la différence $\epsilon_i = x_i - x_{i-1}$.

Les techniques de prédiction appelées DPCM (pour *modulation différentielle d'impulsions codées*), reprennent ce principe et viennent s'insérer dans le processus de compression qui comprend déjà un quantificateur et un codeur.

La description schématique en est donnée en figure 3.27. La différence ϵ_t mesure l'écart entre la donnée x_t et la prédiction $\tilde{x}_t$ retournée par le module P qui la calcule

grâce aux données x_i $(i < t)$ précédemment mémorisées. L'indice de quantification $q_\epsilon = Q(\epsilon_t)$, envoyé par l'émetteur, correspond à la valeur quantifiée $\hat{\epsilon}_t = Q^{-1}(q_\epsilon)$ de la différence.

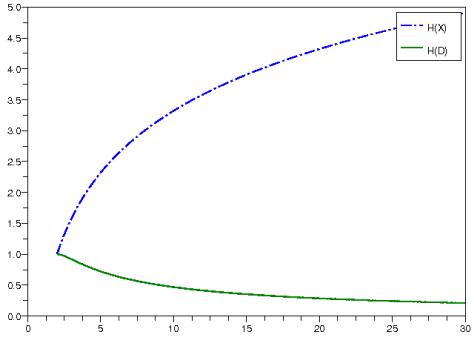


Figure 3.26. *Comparaison des entropies de la source (en pointillée) et de la différence (en trait plein)*

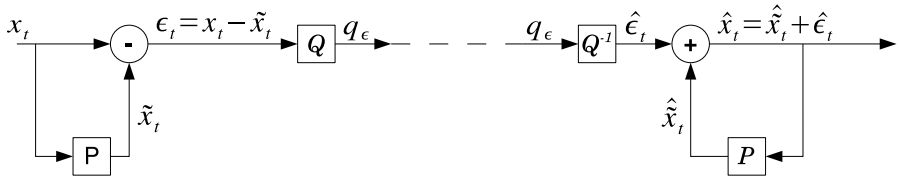


Figure 3.27. *Schéma de codification avec prédiction*

Le récepteur fonctionne à l'inverse. Après déquantification, le signal $\hat{x}_t$ est reconstruit à l'aide de la différence quantifiée $\hat{\epsilon}_t$ et de la prédiction *reconstruite* $\hat{\tilde{x}}_t$. La prédiction est souvent linéaire, représentable matriciellement. La taille de la matrice est fonction du nombre de prédécesseurs pris en compte. Les coefficients de la matrice peuvent être estimés grâce aux premières valeurs à coder ou grâce à un banc d'essais.

Pour l'exemple en début de section, la prédiction est simplement la donnée précédente x_{t-1}. Le schéma étant récurrent, il faut lui trouver une valeur initiale pour démarrer. La technique d'initialisation présentée dans l'exemple est souvent utilisée : les premières données sont transmises sans être modifiées. Le récepteur doit donc être en accord avec l'émetteur sur le nombre de données servant d'initialisation.

Il faut toutefois, faire attention à la quantification, car elle engendre des pertes irréversibles. Ces pertes distordent les valeurs reconstruites et les prédictions : par effet de bord, les distorsions des prédictions vont en s'accumulant puisqu'elles proviennent,

à la fois, de la différence quantifiée et de la prédiction dans un processus récurrent :

$$\hat{x}_t = \widehat{\widehat{x}}_t + \hat{\varepsilon}_t = (\widehat{\widehat{x}}_{t-1} + \underbrace{\hat{\varepsilon}_{t-1}}_{\text{erreurs cumulées}}) + \hat{\varepsilon}_t = \dots$$

Pour éviter ce problème, l'émetteur simule la reconstruction du signal faite par le récepteur. Cette reconstruction lui sert à calculer la prédiction. Puis, il calcule la différence, ϵ_t, entre la donnée x_t, et la prédiction reconstruite $\widehat{\widehat{x}}_t$. Le récepteur récupère cette différence quantifée $\hat{\epsilon}_t$, et l'ajoute à la prédiction $\widehat{\widehat{x}}_t$, pour retrouver le signal $\hat{x}_t$.

On constate alors que les distorsions entre la source x_t, et sa version reconstruite $\widehat{\widehat{x}}_t$, ne sont plus dues qu'à la quantification de la différence[20]. Le schéma modifié de la prédiction est donné en figure 3.28.

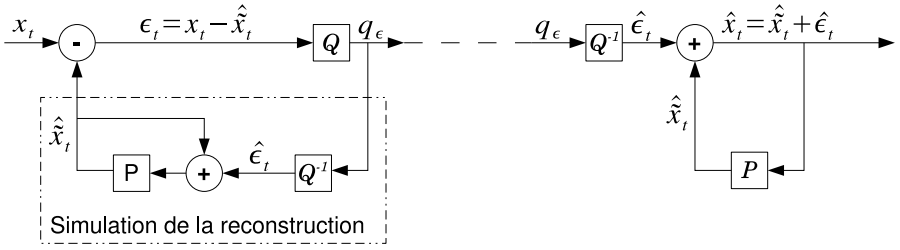


Figure 3.28. *Schéma de codification avec prédiction*

La prédiction est un outil efficace et facile à mettre en œuvre. On la retrouve dans tous les formats et standards de compression des prochains chapitres. Elle est utilisée pour la compression audio (voir chapitre 4), mais aussi, pour la compression des images et des vidéos. Elle est généralisée par les standards MPEG1, MPEG2 et MPEG4 pour l'estimation de mouvement dans les vidéos (voir chapitres 5, 6 et 7).

3.6. Synthèse

Ce chapitre vient de présenter les outils de création de flux binaires : la numérisation et la compression. La numérisation converti un signal analogique (continu) en un signal numérique (discret). Son objectif, tout d'abord orienté télécommunication, a vite débordé sur des domaines aussi divers que l'informatique, l'ergonomie ou encore la conception (design).

Tout signal analogique, c'est-à-dire défini par des valeurs continues, doit être décrit par des valeurs discrètes, dès qu'il s'agit de l'enregistrer sur un support informatique

20. Si nous omettons les perturbations provenant du canal de transmission.

ou de le transmettre sur un réseau tel qu'Internet. Ce travail de numérisation comporte deux processus : l'échantillonnage et la quantification. L'échantillonnage sélectionne les valeurs qui permettent par la suite de reconstruire le signal analogique. Comme le stipule le théorème de Nyquist-Shannon, le pas d'échantillonnage est borné inférieurement par la fréquence maximale du signal source. En deçà de ce seuil, il y a un risque de recouvrement spectral.

La numérisation n'effectue pas que l'échantillonnage. Les valeurs des échantillons, qui sont continues, doivent être quantifiées afin de les coder sur une machine. Le procédé consiste à réduire la taille de l'alphabet initial. Le plus simple est de découper l'alphabet source suivant sa distribution.

Dans le cas d'une distribution uniforme, la quantification uniforme est optimale. Malheureusement, la distribution de l'alphabet source est rarement uniforme. Aussi, faut-il soit la rendre uniforme, soit changer de procédé.

Si le modèle de la distribution est supposé connu, la technique de compansion est adaptée. Elle fait subir une déformation à l'aphabet qui a tendance à uniformiser sa distribution. Cette technique est utilisée pour la compression de la parole.

Mais, si la forme de la distribution n'est pas connue, mis à part qu'elle soit élevée pour les valeurs extrêmes, la quantification à zone morte peut alors être utilisée. Elle permet d'avoir, à la fois, un pas de quantification fin pour les valeurs extrêmes et un pas plus important pour les valeurs moyennes.

Dans d'autres cas, comme les images, la quantification est une matrice de coefficients qui pondère les valeurs observées. Ces pondérations sont choisies de telle manière que le résultat de la quantification soit le moins perceptible possible.

Si, en revanche, les échantillons sont fortement correllés entre eux, une quantification vectorielle s'avère plus adaptée. Les échantillons sont regroupés en vecteurs. Un dictionnaire de vecteurs représentatifs est utilisé pour opérer la quantification.

Une fois le signal numérisé, sa taille trop importante empêche qu'il soit stocké sur un support informatique ou transmis sur un réseau. Il faut, alors, le compresser.

La première des techniques de compression est de reprendre la quantification afin de réduire encore la taille de l'alphabet. Cependant, cette technique est limitée par la fidélité voulue lors de la reconstruction du signal.

L'amélioration de la compression, tout en contrôlant la fidélité, est possible avec les codeurs entropiques. L'objectif est similaire à la quantification : réduire la taille de l'alphabet source. Pour y parvenir, ces techniques éliminent les redondances du signal original. L'information essentielle – c'est-à-dire suffisante et nécessaire à la reconstruction du signal – est indiquée par l'entropie du signal source :

– les codeurs par plages repèrent les séquences de valeurs identiques pour les transformer en couples <*valeur, cardinalité*> ;

– les codeurs à dictionnaires dynamiques construisent un dictionnaire au fur et à mesure qu'ils lisent la source. Simultanément, ils utilisent les mots de ce dictionnaire afin de réduire la taille du signal original ;

– les codeurs à longueurs variables, comme les codeurs de Huffman, de Golomb ou arithmétiques, se fondent sur une étude statistique de la distribution de l'alphabet du signal à coder. Connaissant la distribution du signal source, le codeur peut définir le type de code (binaire, unaire, etc.) et la longueur optimale du code pour chacune des valeurs de l'alphabet source ;

– le codage par prédiction est une technique simple, mais, efficace. Partant du principe que tout signal est corrélé, des prédictions peuvent être faites à partir des éléments déjà codés. Si ces prédictions sont bonnes, la différence entre les valeurs originales et les prédictions sont faibles. Il est alors plus intéressant de coder ces différences que les échantillons originaux. Il faut simplement s'assurer que le décodeur utilise la même technique de prédiction que le codeur.

Tous ces outils de quantification et de codage sont largement utilisés pour la compression de l'audio, de l'image et de la vidéo. Toutefois, les capacités sensorielles de l'humain ne sont pas prises en compte. Le chapitre suivant montre qu'une étude des modèles auditif et visuel permet d'augmenter encore la compression, sans pertes significatives pour l'utilisateur.

Chapitre 4

Perception

Ce chapitre[1] est consacré aux modèles communément utilisés pour décrire la vue et l'ouïe. Mais, pourquoi s'intéresser seulement à ces deux sens ? Bien que le toucher, le goût et l'odorat soient également importants chez l'humain, les données multimédias actuelles reposent sur les supports informatiques classiques que sont l'écrit, le son et la visualisation. L'incorporation d'autres sens reste encore du domaine de la recherche en *interaction humain machine* (IHM[2]).

Les sens, quels qu'ils soient, sont des perceptions subjectives de l'environnement. Cette subjectivité, si elle est prise en compte, permet d'augmenter la compression des données multimédias avec des détériorations imperceptibles ou, à défaut, les moins génantes possibles.

La section 4.1 décrit la physiologie de la perception visuelle humaine et la définition physique de la couleur. Il s'agit du modèle de Ewald Heiring pour la perception humaine et du modèle de Thomas Young pour la physique. Bien que longtemps considérés antagonistes, ces deux modèles sont, en fait, complémentaires. De cette constatation découlent plusieurs modèles de représentation de la couleur, en relation les uns avec les autres. Les principaux sont expliqués, ainsi que leurs raisons d'être.

La section 4.2 présente les techniques de compression audio. Les techniques usuelles, telle que la prédiction, sont tout à fait adaptées à cette tâche, compte tenu de la forte corrélation temporelle du son. Toutefois, ces techniques délivrent des signaux sonores requérant des débits assez élevés. Aussi, d'autres techniques, dérivées des

1. Ce chapitre est dédicace à Juliette et Romane.

2. en anglais CHI : *Computer Human Interaction*.

propriétés du signal sonore et du système auditif humain, permettent d'augmenter les taux de compression. Celles-ci assurent un codage du son de qualité professionnelle pour des débits faibles et variables.

4.1. Les espaces de couleurs

L'*œil*, ou *globe oculaire*, présenté en figure 4.1, est une sphère creuse dont la surface extérieure protectrice (le blanc de l'œil) est appelée la *sclérotique*. Il est rempli d'un liquide appelé l'*humeur vitrée*. Dans la partie antérieure de l'œil, la sclérotique est prolongée par la *cornée*. La lumière peut alors passer. Derrière la cornée, l'*humeur acqueuse* est le liquide qui sépare le *cristallin* de la cornée. Le cristallin, aidé de l'*iris*, permet d'accomoder la vision à l'intensité lumineuse et à la distance des objets observés. L'iris s'ouvre et se ferme tel un diaphragme d'appareil photographique afin de contrôler le flux lumineux entrant. Le cristallin est une lentille biconvexe. Pour une vue de loin, le cristallin est en position de repos. Les rayons lumineux sont quasi parallèles. La réfraction de l'humeur acqueuse assure la convergence des rayons lumineux pour une projection correcte de l'image sur la rétine. Cette convergence, identifiée par le centre optique où tous les rayons lumineux passent, doit toujours être la même.

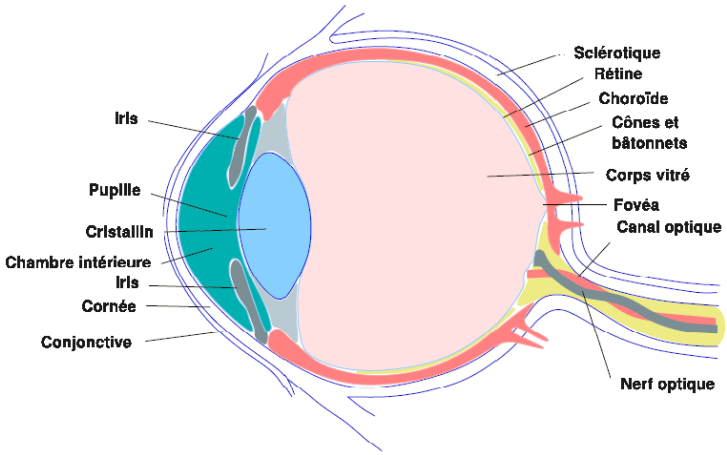


Figure 4.1. *Anatomie de l'œil*

Lorsque l'œil perçoit des objets à moins de six mètres, les rayons ne sont plus parallèles. Le cristallin doit s'épaissir pour permettre la convergence vers le centre optique. La figure 4.2 illustre le cas de la vue de loin avec un cristallin au repos et celui de la vue de près avec un épaississement du cristallin.

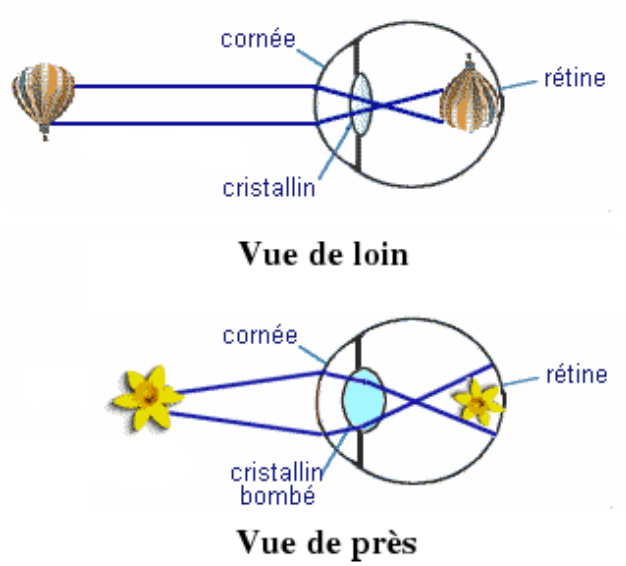


Figure 4.2. *Contraction du cristallin pour une adaptation à la vue de près (grâcieusement communiquée par Ophtasurf : http://ophtasurf.free.fr)*

Dans sa partie intérieure, au niveau de la sclérotique, l'œil est recouvert par une membrane noire, la *choroïde*, qui est le manteau vasculaire de l'œil. Puis, plus à l'intérieur, se trouve la *rétine* constituée de deux couches. La couche *pigmentaire*, la plus externe, absorbe la lumière afin d'en limiter la diffusion dans l'œil. La couche interne est le siège de la perception grâce à des neurones photosensibles. Ces neurones se distinguent en bâtonnets et cônes, suivant qu'ils permettent la vue de jour ou de nuit.

Les *bâtonnets* permettent une vision nocturne à cause de leur grande sensibilité[3]. Il s'ensuit qu'ils sont vite saturés. Ils n'envoient donc plus d'informations dès que l'intensité lumineuse augmente légèrement. Cette vision nocturne est *achromatique*[4].

En revanche, les *cônes* sont fort bien adaptés à la vision de jour. En opérant comme des filtres fréquentiels, ils permettent la perception de la couleur. Leurs bandes fréquentielles définissent trois plages de couleurs :

1) les cônes L détectent les *grandes ondes* qui caractérisent les teintes rouges; ils sont identifiés par la lettre L pour *long wave cones* ;

3. Un bâtonnet réagit dès qu'un seul photon le rencontre.

4. Ce qui explique que « la nuit, tous les chats sont gris ».

2) les cônes M détectent les *ondes moyennes* qui caractérisent les teintes vertes; ils sont identifiés par la lettre M pour *medium wave cones* ;

3) les cônes S détectent les *ondes courtes* caractérisent les teintes bleutées; ils sont identifiés par la lettre S pour *short wave cones*.

La figure 4.3 montre la *sensibilité spectrale* des cônes en traits continus et celles des bâtonnets en traits pointillés. La courbe de la sensibilité spectrale des cônes S (la plus à gauche sur le graphique) couvre la plage des longueurs d'ondes allant de $\approx 370\,\mathrm{nm}$ à $\approx 470\,\mathrm{nm}$. Autrement dit, les cônes S absorbent tous les photons émis dans cette plage de fréquences. Le raisonnnement est le même pour les cônes M et L. Les valeurs minimale et maximale de ces trois sensibilités spectrales définissent le *spectre du visible*.

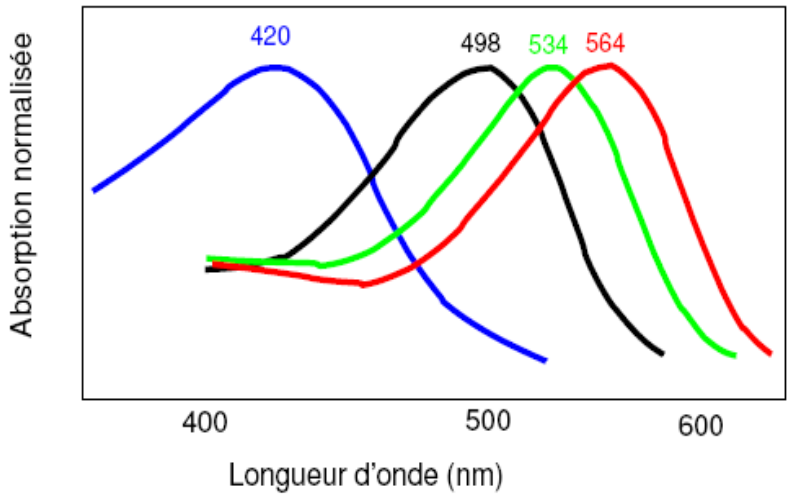


Figure 4.3. *Les bandes passantes rouge (420 nm), vert (534 nm) et bleu (564 nm) des cônes. La réponse des bâtonnets est maximale pour $\lambda = 498$ nm*

Ce spectre est relativement restreint puisque compris entre ≈ 400 nm et ≈ 700 nm (voir figure 4.4). A chaque couleur dans ce spectre, correspond une onde pure définie par sa longueur. Ainsi le bleu est défini par une longueur d'onde $\lambda_b = 420\,\mathrm{nm}$, le vert par une longueur d'onde $\lambda_v = 534\,\mathrm{nm}$ et le rouge par une longueur d'onde $\lambda_r = 564\,\mathrm{nm}$. Cette étude a permis au physicien *Thomas Young* de construire le modèle tridimensionnel de représentation des couleurs : le modèle RVB pour rouge vert bleu[5].

5. En anglais : RGB pour *Red Green Blue*.

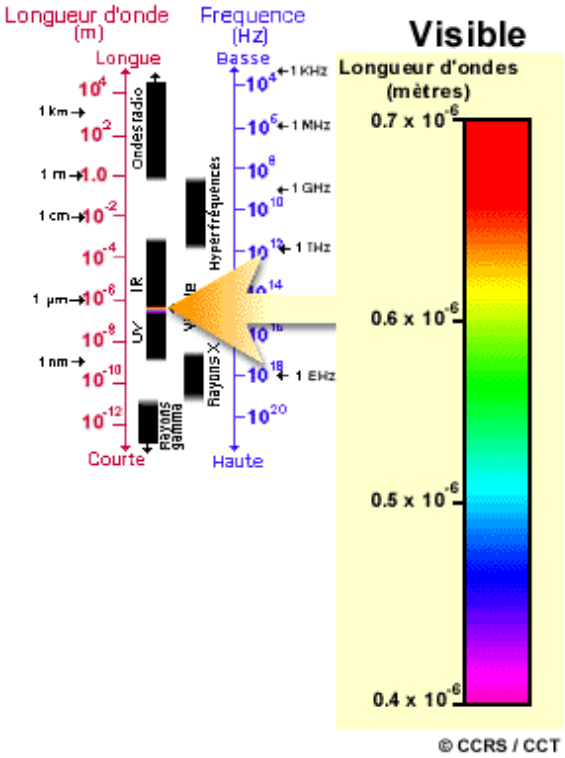


Figure 4.4. *Le spectre du visible*
(grâcieusement communiquée par le centre canadien de télédétection : http://ccrs.nrcan.gc.ca)

Comme l'illustre la figure 4.5, les cônes et les bâtonnets ne sont pas uniformément répartis. Les bâtonnets sont pratiquement sur toute la surface de la rétine, alors que les cônes sont principalement situés dans une zone appelée *macula*. Cette région, petite en taille, ne contient aucun bâtonnet. Elle concentre une très forte densité de cônes, apportant une vision précise de notre environnement direct. La macula possède, elle-même, une région de densité plus importante, appelée *fovea*. La fovea permet au cerveau d'apprécier les formes et les dimensions des objets, ainsi que leurs distances à l'observateur.

Par ailleurs, des études en psychologie ont montré que le modèle RVB n'est pas celui utilisé par les zones supérieures du système visuel humain. Le cerveau utilise une mesure relative et non absolue des couleurs. Ainsi, le psychologue *Ewald Heiring* proposa un modèle, dit antagoniste, tridimensionnel où chaque axe définit une opposition entre deux couleurs. Un axe définit l'antagonisme entre le blanc et le noir, un autre l'antogonisme entre le rouge et le vert et un troisième entre le bleu et le jaune.

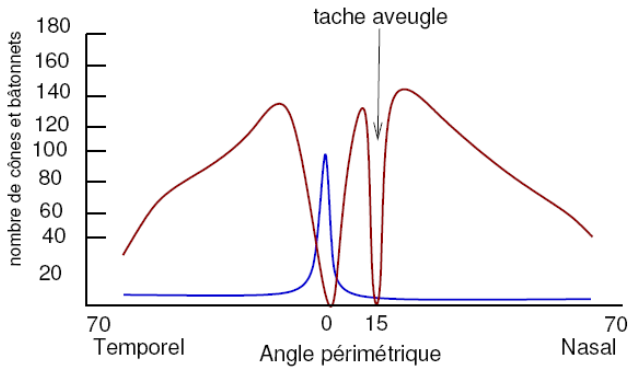


Figure 4.5. *Répartition des cônes et des bâtonnets sur la surface de la rétine. Les distances sont angulaires avec comme origine, le centre optique et comme angle nul (0 stéradiant), la direction de vue qui exprime la position centrale d'observation de l'œil (voir figure 4.1). Cette direction passe par le centre du cristallin, le centre optique et la fovea.*

Bien qu'*a priori* surprenant, ces modèles ne se contredisent pas. Une étude plus approfondie de la rétine, indique que sa couche interne n'est pas uniquement constituée de photorécepteurs. Elle contient d'autres neurones qui viennent regrouper les informations des photorécepteurs. Il existe, en fait, sept niveaux hiérarchiques de neurones; chaque niveau répondant à un type de regroupement différent. Ces neurones communiquent à l'aide de prolongements, appelé *axones*. Il est étonnant de constater que tous ces neurones viennent se positionner entre la lumière et les photorécepteurs (voir figure 4.6).

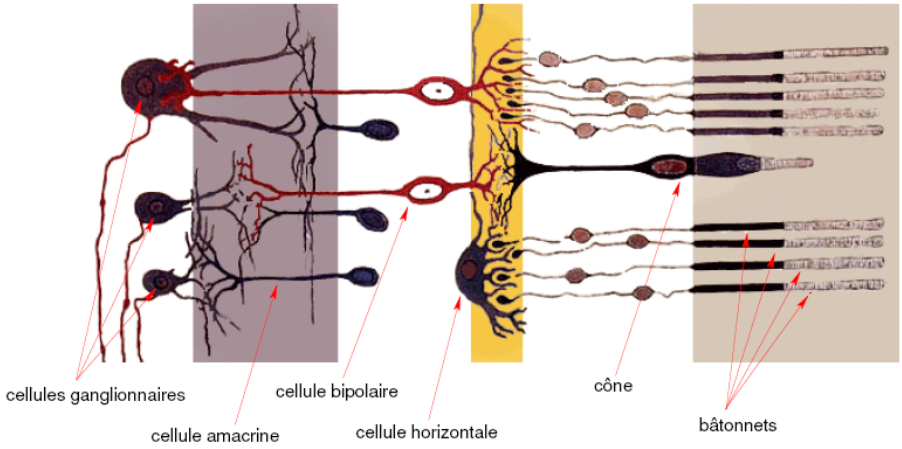


Figure 4.6. *Hiérarchie des neurones assurant la mise en forme de l'information perçue par les cônes et les bâtonnets*

Les neurones les plus internes, appelés *cellules ganglionaires*, regroupent l'information en dernier. L'ensemble des axones des cellules ganglionaires se regroupent pour former le nerf optique. Ce regroupement définit une zone sans photorécepteur, donc aveugle, appelée la *papille optique* (voir figure 4.1).

Le nerf optique va jusqu'aux *corps genouillés latéraux* (CGL), gauche et droit, des deux hémisphères du cerveau. A ce niveau, l'information est ordonnée pour être ensuite transmise au cortex visuel primaire (zone V1). Le cortex effectue les premiers traitements de l'information visuelle où les propriétés de mouvement, de forme, de texture et de couleur sont extraites. Il est à noter que le mouvement et la forme sont des informations achromatiques. Ainsi, la couleur, apparaît comme secondaire pour la perception humaine.

Ce très bref aperçu de l'anatomie de l'œil, qui peut être complété par la lecture du livre de David Hubel [HUB 94], apporte déjà des informations intéressantes.

On retiendra principalement que la perception humaine se concentrerait sur des informations achromatiques que la couleur semble venir confirmer en second plan. Ce trait particulier de la vision humaine a été utilisé pour définir des espaces tridimensionnels de représentation des couleurs, avec un axe achromatique et deux axes chromatiques. Ces modèles sont similaires, dans le principe, au modèle proposé par Ewal Heiring.

La suite de cette section va se poursuivre par une étude physique de la couleur. Au paragraphe 4.1.1, quelques définitions, comme celle de la couleur blanche, celle des couleurs primaires et secondaires et celles des synthèses possibles, sont données. Ensuite, le paragraphe 4.1.3 introduit le modèle de référence de couleurs générique. A partir de ce modèle, le paragraphe 4.1.4 fournit un ensemble non exhaustif des modèles de couleurs couramment utilisés. Elle précise pour quel type d'applications chacun d'eux a été défini.

4.1.1. *Définitions physiques de la couleur*

La couleur blanche a été étudiée expérimentalement par Isaac Newton. Il a démontré qu'elle était une combinaison du spectre des couleurs. Pour cela, il a orienté un faisceau de lumière blanche (la lumière du jour) vers un prisme qui a décomposé cette lumière en un ensemble de couleurs identiques à celles d'un arc-en-ciel. Puis, il a positionné un autre prisme face à un des faisceaux de ce spectre pour démontrer que la décomposition est terminale. La figure 4.7 schématise l'expérience. Les couleurs du spectre ne pouvant être décomposées en d'autres couleurs, elles sont dites *monochromatiques*. La couleur blanche, tout comme la couleur noire, n'existe qu'à travers elles. La couleur noire est l'absence de toute onde monochromatique. Elle est donc non observable. La lumière blanche, à l'opposé, couvre le spectre du visible.

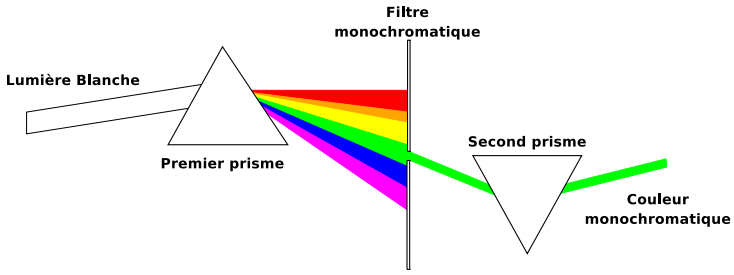


Figure 4.7. *L'expérience d'Isaac Newton*

De plus, Isaac Newton a eu l'idée de placer les couleurs monochromatiques, dans l'ordre du violet au rouge, sur un *cercle chromatique* (voir figure 4.8). Ce cercle met l'accent sur les *couleurs complémentaires* qui sont diamètralement opposées sur le cercle. Cette complémentarité implique que le fusionnement d'un faisceau lumineux monochromatique, avec un faisceau de sa couleur complémentaire, produit la lumière blanche.

Cette fusion lumineuse est appelée *synthèse additive*. La *synthèse soustractive* correspond au phénomène de filtrage d'une source lumineuse blanche par les pigments d'une surface. Ainsi, le mélange de deux pigments de couleurs complémentaires réfléchit une onde grise lorsqu'il est éclairé par une source blanche. Pour des pigments sans défaut de chromaticité – pigments théoriques – la source serait totalement absorbée et on percevrait l'absence de couleur : le noir (voir figure 4.8).

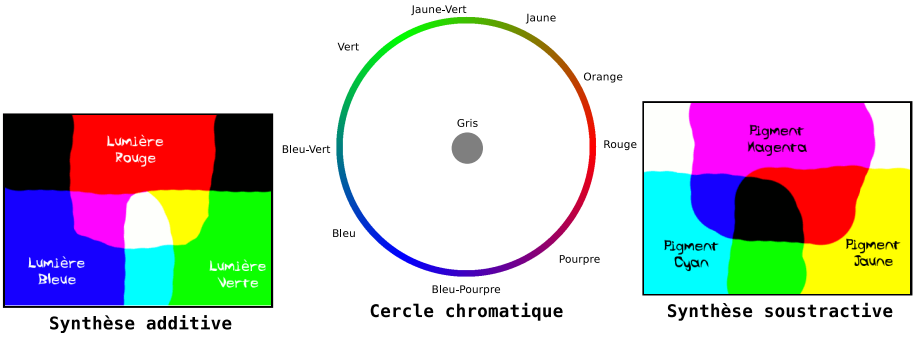


Figure 4.8. *Le centre du cercle chromatique est la valeur de gris correspondant à l'intensité lumineuse des couleurs de ce cercle. Cette valeur peut varier du blanc au noir. La synthèse additive correspond à la fusion de faisceaux lumineux de couleurs différentes. Elle est illustrée par le diagramme de gauche avec les faisceaux lumineux des trois couleurs rouge, vert et bleu. La synthèse soustractive correspond au filtrage qu'opèrent les pigments d'une surface sur la lumière blanche. Elle est illustrée par le diagramme de droite avec les pigments des trois couleurs cyan, jaune et magenta.*

La synthèse additive est utilisée en infographie pour l'affichage sur les écrans alors que la synthèse soustractive est utilisée en imprimerie.

Un peu plus tard, Thomas Young observe que la synthèse additive de l'ensemble des couleurs monochromatiques produit la lumière blanche. Il constate même que trois couleurs, qu'il appellera *primaires*, équidistantes entre elles sur le cercle chromatique, suffisent pour retrouver la lumière blanche. Parmi ces triplets de primaires, le plus connu est le <*rouge, vert, bleu*>. Les *couleurs secondaires* sont définies comme étant au milieu des primaires sur le cercle chromatique. Par exemple, le jaune est à la même distance du rouge et du vert.

Par la suite, James Clerck Maxwell transforme le cercle chromatique en un triangle équilatéral dont les sommets sont les primaires rouge, vert et bleu. Les couleurs sont maintenant définies par rapport à leurs coordonnées dans ce triangle. Le modèle correspond à celui de la synthèse additive : tout point à l'intérieur du triangle est une couleur obtenue par composition des trois primaires. Les coordonnées (α_r α_v α_b) d'un point du triangle sont alors définies par rapport aux sommets du triangle. α_r (resp. α_v et α_b) est le pourcentage de rouge (resp. de vert et de bleu) dans la couleur. Ainsi, le barycentre de coordonnées ($\frac{1}{3}$ $\frac{1}{3}$ $\frac{1}{3}$) désigne le blanc. La couleur opposée au rouge est de coordonnées (0 0,5 0,5). La figure 4.9 montre ce triangle inscrit dans le cercle chromatique.

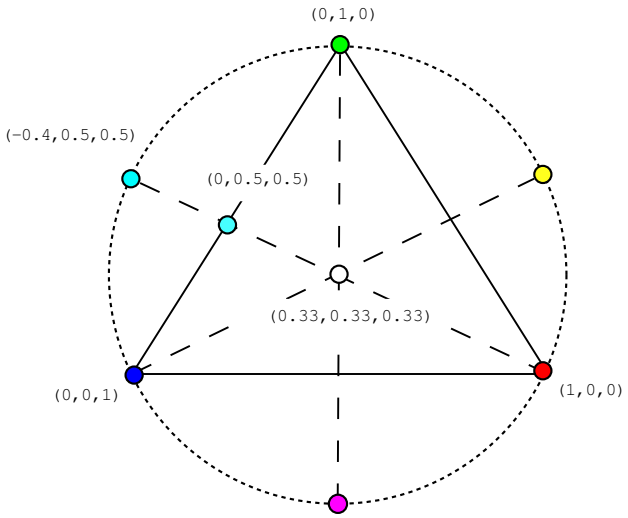


Figure 4.9. *Le triangle de Maxwell*

Mais, certaines couleurs se trouvent être à l'extérieur de ce triangle. Par exemple, dans la figure 4.9, si l'on trace la demi-droite partant de la couleur rouge, et passant par

le blanc, on intersecte le triangle au milieu du coté opposé. La couleur de coordonnées (0 0,5 0,5) n'est pas le complémentaire du rouge. Celui-ci est défini diamètralement opposé au rouge sur le cercle chromatique. Il a pour coordonnées (−0,4 0,5 0,5). Bien qu'additif, on constate que le modèle est obligé d'introduire des coordonnées négatives pour caractériser toutes les couleurs possibles. En procédant ainsi, pour toutes les couleurs monochromes, James Clerck Maxwell identifie les coordonnées des couleurs monochromatiques dans le repère du triangle. Comme l'illustre la figure 4.10a, les primaires rouge, vert et bleu sont les seules couleurs monochromatiques définies dans le triangle de Maxwell.

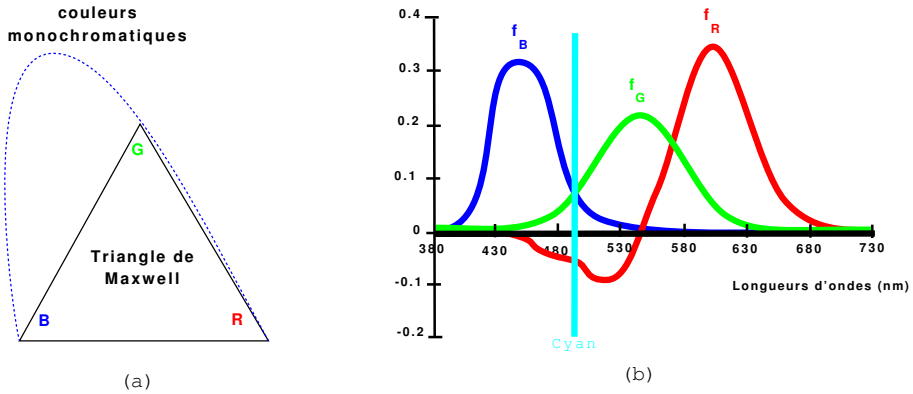


Figure 4.10. *(a) Définition des couleurs monochromatiques par rapport au triangle de Maxwell. (b) Fonctions colorimétriques des primaires R, G et B.*

4.1.2. *L'espace CIE RGB*

La représentation de Maxwell permet d'observer de nouvelles propriétés des couleurs : toute demi-droite partant du barycentre définit une suite de couleurs qui va du blanc jusqu'à la couleur monochromatique portée par cette demi-droite. Toutes ces couleurs ont donc la même *teinte*, indiquée par la couleur monochromatique de la demi-droite. Elles se distinguent, entre elles, par leurs *saturations*, allant en augmentant, quand elles s'éloignent du blanc.

De plus, l'ensemble des coordonnées des couleurs monochromatiques permet de tracer les *fonctions colorimétriques* des primaires rouge, vert et bleu (voir annexe C.1). Comme le montre la figure 4.10b, pour chaque couleur monochromatique du spectre, les fonctions colorimétriques fournissent les pourcentages de chacune des primaires. Par exemple, le cyan – qui est le complémentaire du rouge – correspond à un mélange équitable de vert et de bleu, auquel il faut soustraire du rouge.

Du fait de l'ambiguïté de la définition de la couleur blanche (voir paragraphe 4.1.3), il existe de nombreux espaces RGB. Mais, l'espace *RGB* introduit par le Comité international de l'éclairement (CIE) en 1931 est le standard communément admis. Celui-ci est défini à partir de trois primaires R, G et B comme le montre la figure 4.11.

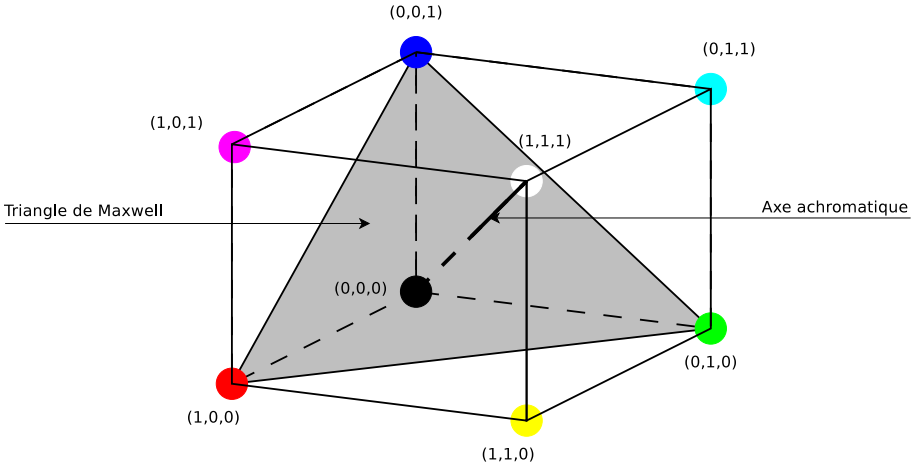


Figure 4.11. *L'espace CIE RGB*

Les longueurs d'ondes associées à chacune des primaires R, G et B sont, respectivement, 700 nm, 546,1 nm et 435,8 nm. La puissance de chacune de ces primaires est ajustée de façon à obtenir un triplet de coordonnées identiques pour toute couleur dont le spectre est d'égale énergie. Ainsi, l'origine du repère est le noir, le point unité est le blanc et tous les points entre le noir et le blanc sont les nuances de gris donc *achromatiques*.

Le triangle de Maxwell est dans le plan unitaire. Ses sommets sont les vecteurs de la base, c'est-à-dire les primaires R, V et B. Il est souvent fait référence aux coordonnées CIE RGB *normalisées* qui sont considérées indépendantes de la luminosité dans la plupart des applications [GER 06] :

$$\begin{cases} r &= \frac{R}{R+G+B} \\ g &= \frac{G}{R+G+B} \\ b &= \frac{B}{R+G+B} \end{cases}$$

REMARQUE 4.1.– *Classiquement, quel que soit le modèle RGB utilisé, les valeurs sont supposées être comprises entre* 0 *et* 1 *afin d'assurer toute manipulation mathématique de la couleur.*

4.1.3. *L'espace CIE XYZ*

Le triangle de Maxwell permet d'avoir des coordonnées associées aux couleurs. Cependant, ce système additif est obligé de prendre en compte des pondérations négatives. Pour éviter ce problème, le modèle *XYZ* a été défini tel que :

1) la synthèse additive soit effective et totale : pas de pondération négative ;

2) la primaire Y corresponde à la luminosité[6] et que les deux autres primaires définissent la *chrominance* ;

3) le blanc soit la valeur d'équi-énergie : $\alpha_X = \alpha_Y = \alpha_Z$.

La figure 4.12 montre les fonctions colorimétriques obtenues par une transformation contrainte des fonctions colorimétriques du modèle CIE *RGB*.

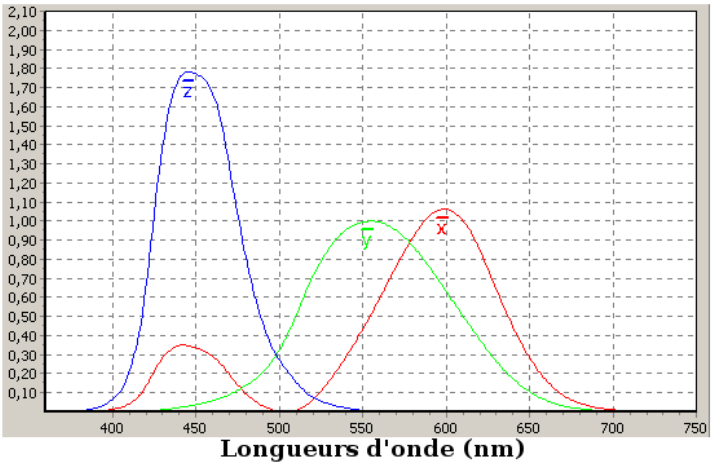


Figure 4.12. *Fonctions colorimétriques du modèle XYZ*
(grâcieusement communiquée par Earl F. Glynn : http://www.efg2.com/)

La luminosité Y a été modélisée de façon à approximer la *fonction d'efficacité lumineuse*. Cette dernière est estimée expérimentalement suivant un principe similaire à l'estimation des fonctions colorimétriques : les couleurs monochromatiques sont présentées deux par deux avec la même luminance au cobaye humain. Celui-ci indique laquelle des deux lui paraît la plus lumineuse.

6. On fera la distinction entre luminosité et luminance. La luminance correspond à la quantité d'énergie qu'une source émet ou qu'un objet réfléchi alors que le luminosité est la perception que l'on a de la luminance.

Comme le montre la figure 4.13, la perception la plus intense est celle du jaune, alors qu'elle diminue pour les extrémités (le bleu et le rouge) du spectre visible. Un simple test à l'aide d'un logiciel graphique permet de le constater.

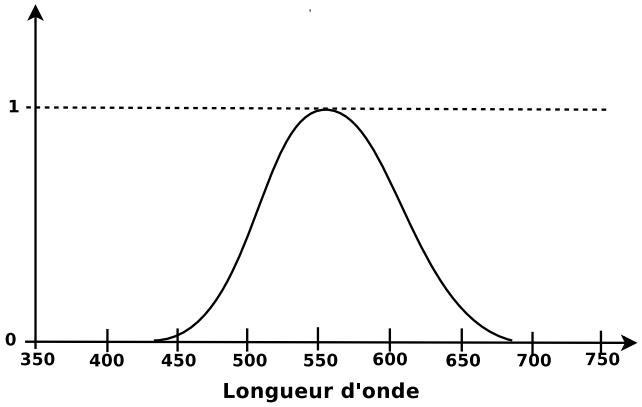


Figure 4.13. *La fonction d'efficacité lumineuse*

Les trois primaires X, Y et Z sont alors des valeurs spectrales non visibles. L'espace tridmensionnel XYZ contient donc l'ensemble des couleurs monochromatiques. L'intersection de ces dernières avec le plan unitaire – aussi appelé plan chromatique – définit la courbe de saturation maximale des couleurs. On en déduit que toute demi-droite partant de l'origine (la couleur noire) dont l'intersection avec le plan chromatique se situe à l'interieur de la courbe de saturation maximale, définit des couleurs de chromaticité constante (même teinte et même saturation); seule l'intensité des couleurs sur cette demi-droite varie (voir figure 4.14). En particulier, la demi-droite passant par le point unité définit les couleurs d'équi-énergie, c'est-à-dire les niveaux de gris qui vont du noir – de coordonnées (0 0 0) – au blanc situé dans le plan chromatique – de coordonnées (1/3 1/3 1/3).

La projection des valeurs du plan chromatique sur le plan défini par les axes X et Y, fournit le *diagramme xy* de la figure 4.15. Ceci revient à normaliser les coordonnées dans l'espace XYZ :

$$\begin{cases} x &= \frac{X}{X+Y+Z} \\ y &= \frac{Y}{X+Y+Z} \\ z &= \frac{Z}{X+Y+Z} \end{cases}$$

et à ne conserver que deux valeurs; la troisième se déduit des valeurs des deux premières. Puisqu'il y a projection, il y a perte d'information : où est le noir, où est le gris ? La luminosité n'est pas présente dans de ce diagramme.

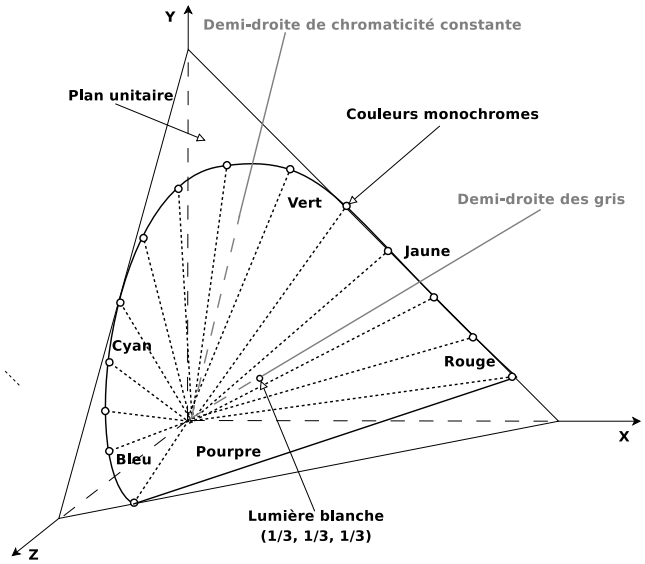


Figure 4.14. *Toutes les couleurs sont définies dans le premier octan de l'espace tridimensionnel CIE XYZ*

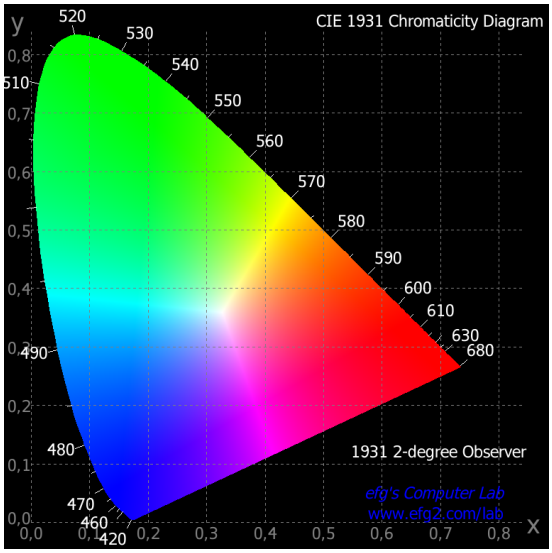


Figure 4.15. *Diagramme xy*
(grâcieusement communiquée par Earl F. Glynn : http://www.efg2.com)

4.1.3.1. *Transformation entre le modèle XYZ et un modèle RVB*

La transformation est linéaire. Elle est décrite par la matrice de passage P :

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = P \begin{pmatrix} R \\ V \\ B \end{pmatrix}$$

$$\begin{pmatrix} R \\ V \\ B \end{pmatrix} = P^{-1} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

Les coefficients de P dépendent du blanc de référence choisi. Il existe donc plusieurs solutions pour la matrice de passage. La section suivante fournit plusieurs valeurs standards de blancs.

4.1.3.2. *Limite de l'espace XYZ*

Ce modèle propose un espace de représentation des couleurs muni d'un repère dans lequel deux couleurs distinctes ont des coordonnées différentes et positives. Malheureusement, les études de Mc Adam montrent que la perception humaine des couleurs n'est pas linéaire. Certaines couleurs paraissent visuellement identiques, alors qu'elles sont physiquement distinctes. *A contrario*, d'autres paraissent visuellement très différentes, alors que leurs coordonnées dans le diagramme *xy* sont relativement proches. Mc Adam a visualisé ses expériences sur le diagramme *xy* en plaçant des ellipses au sein desquelles les couleurs ne sont pas visuellement différenciables. La figure 4.16 visualise l'ensemble des ellipses, appelées *métamères*.

La perception visuelle n'est donc pas linéaire contrairement au modèle *XYZ*. Cependant, ce modèle permet principalement de définir un espace vectoriel pour la synthèse additive. Il permet également de construire d'autres modèles de couleurs plus adaptés à certains types d'applications. Nombre de ces modèles partage la propriété de séparation de la luminosité et de la chrominance et les notions d'opposition des couleurs *rouge-vert* et *bleu-jaune*. Ainsi, ils désignent leur axe de luminosité Y.

Par ailleurs, la notion de blanc est une donnée variable avec le contexte. En termes de flux lumineux, il est usuel de considérer la source d'éclairage comme étant le générateur de la couleur blanche. Mais, cette source peut être aussi différente que le soleil, une ampoule au tungstène ou encore une ampoule à incandescence. La perception visuelle de surfaces supposées blanches est liée à cette source d'éclairage. Aussi, plusieurs valeurs de *blancs de référence* sont possibles. La table 4.1 en donne quelques-unes dans le repère *XYZ* et la figure 4.17 présente la place des blancs dans le diagramme *xy*.

On va maintenant présenter certains modèles de couleurs en stipulant leur intérêt.

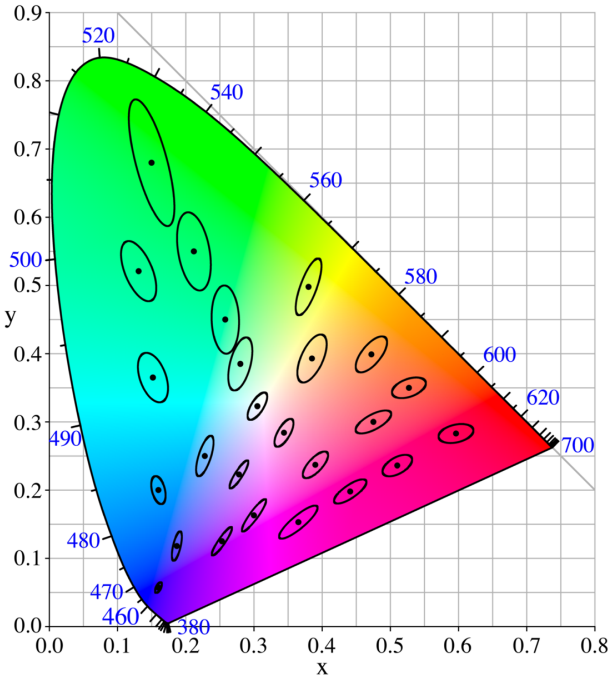


Figure 4.16. *Les métamères résultantes des expérimentations de Mc Adam*

Eclairage	Dénomination	Orientation de l'observateur	X	Y	Z
Tungstène (porté à 2 848 °K)	A	2 °	109,85	100	35,585
		10°	111,144	100	35,2
Ciel nuageux 6 700 °K	C	2 °	98,07	100	118,23
		10°	97,28	100	116,14
Ciel nuageux 6 500 °K	D65	2°	95,047	100	108,883
		10°	94,811	100	107,304
Soleil direct 5 000 °K	D50	2°	96,422	100	82,521
		10°	96,72	100	81,427
Sources fluorescentes	F2	2°	99,186	100	67,393
		10°	103,279	100	69,027

Tableau 4.1. *Description des éclairages standard [CIE 86]. La troisième colonne indique la position de l'observateur par rapport à la cible éclairée lors des expérimentations (le standard définit deux positions possibles)*

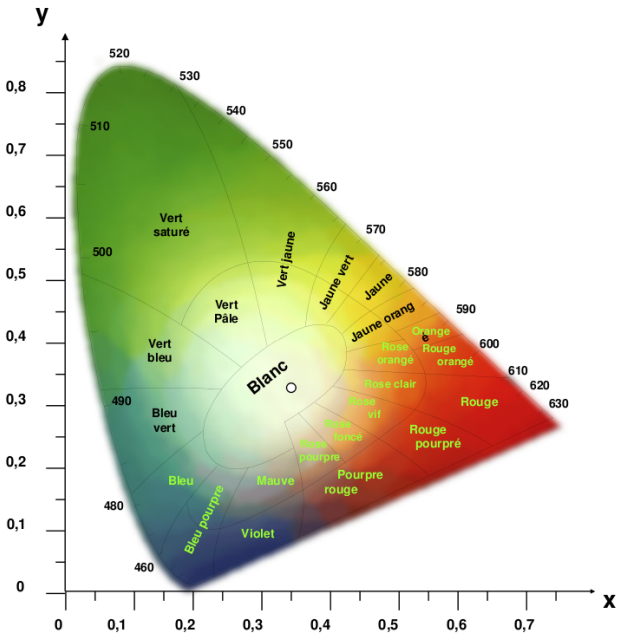


Figure 4.17. *Partitionnement du diagramme xy (grâcieusement communiquée par Jacques Weiss)*

4.1.4. *L'espace CIE* $L^*a^*b^*$

Dans le domaine de la modélisation de la vision humaine, le modèle $L^*a^*b^*$ définit un espace où la notion de distance euclidienne est une approximation considérée correcte de la perception visuelle des couleurs.

Aussi, le passage du modèle *XYZ* au modèle $L^*a^*b^*$ n'est pas linéaire :

$$\begin{aligned} L^* &= \begin{cases} 116 * \left(\frac{Y}{Y_0}\right)^{\frac{1}{3}} - 16 & \textbf{si } \frac{Y}{Y_0} > 0.008856 \\ 903.3 * \left(\frac{Y}{Y_0}\right) & \textbf{sinon} \end{cases} \\ a^* &= 500\left(f\left(\frac{X}{X_0}\right) - f\left(\frac{Y}{Y_0}\right)\right) \\ b^* &= 300\left(f\left(\frac{Y}{Y_0}\right) - f\left(\frac{Z}{Z_0}\right)\right) \end{aligned}$$

avec :

$$f(x) = \begin{cases} x^{\frac{1}{3}} & \textbf{si } x > 0.008856 \\ 7.787x + \frac{16}{116} & \textbf{sinon} \end{cases}$$

et où la composante (X_0, Y_0, Z_0) représente le blanc de référence *choisi au préalable*. La racine cubique simule assez fidèlement le fonctionnement de l'œil humain. Le seuil de 0,008856 est choisi afin d'éviter la pente trop abrupte de la racine cubique dans les faibles valeurs de luminance (les batônnets prennent alors le relais chez l'humain). Pour les fortes luminances, la racine cubique présente une courbure approchant l'effet de saturation de l'œil (voir figure 4.18). Les composantes a et b représentent respectivement l'opposition de couleurs vert-rouge et l'opposition de couleurs bleu-jaune.

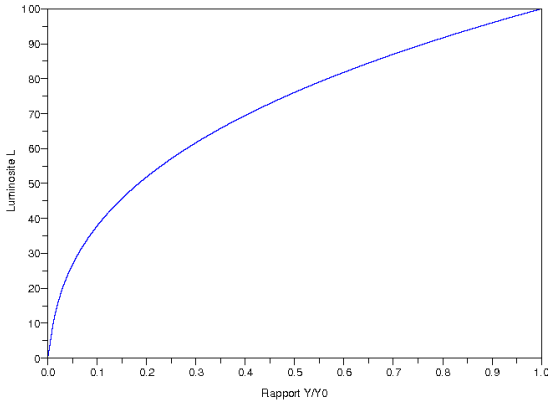


Figure 4.18. *La luminosité L en fonction du rapport* $\frac{Y}{Y_0}$

Les composantes de teinte C_{ab} et de saturation h_{ab} sont alors calculables :

$$\begin{array}{rcl} C_{ab} & = & \sqrt{a^2 + b^2} \\ h_{ab} & = & \arctan\left(\frac{a}{b}\right) \end{array}$$

La différence perceptuelle entre les couleurs est estimée à l'aide de la distance euclidienne :

$$\Delta E_{ab} = \sqrt{\Delta a^2 + \Delta b^2 + \Delta L^2}$$

avec :

$$\left\{ \begin{array}{rcl} \Delta a & = & a_1 - a_2 \\ \Delta b & = & b_1 - b_2 \\ \Delta L & = & L_1 - L_2 \end{array} \right.$$

4.1.5. *Les espaces de manipulation des couleurs*

Ces espaces servent à formaliser le mode d'appréciation des couleurs. Ainsi, ils comportent des axes décrivant une couleur suivant :

1) sa *teinte* : c'est-à-dire sa couleur dominante ;

2) sa *saturation* : est-elle délavée ou bien pure ?

3) son *intensité*, aussi appelée, *valeur*, qui correspond à la luminosité.

Pour cette famille d'espaces, l'axe des niveaux de gris représente la mesure d'intensité lumineuse et la chromaticité est définie par des coordonnées *polaires* dans le plan défini par les deux autres axes. Le module et l'angle du projeté du vecteur couleur dans ce plan correspondent, respectivement, à la saturation et à la teinte. La teinte est donc définie modulo 360°.

Les deux principaux espaces de cette famille sont l'espace TSI[7] : sigle pour teinte, saturation et intensité; et l'espace TSV[8] : sigle pour teinte, saturation et valeur.

4.1.5.1. *L'espace TSI*

C'est une déformation de l'espace *RGB* où l'origine est toujours le noir. L'axe de l'intensité I correspond à la diagonale principale du cube *RGB* allant du noir au blanc. Pour une valeur d'intensité donnée, les couleurs sont comprises dans le cercle chromatique. Au sein de ce cercle les composantes de teinte T, et de saturation S, sont données par les coordonnées polaires. La figure 4.19 visualise cet espace couleur. Les limites dues à l'utilisation du triangle de Maxwell sont encore présentes. Les équations de passage de l'espace RGB[9] vers l'espace TSI sont :

$$\begin{cases} I &= \frac{R+G+B}{3} \\ S &= \frac{I-\min(R,G,B)}{I} \\ T &= \arccos\left(\frac{(R-G)+(R-B)}{2\sqrt{(R-G)^2+(R-B)(G-B)}}\right) \end{cases}$$

avec les contraintes suivantes :

$$\begin{array}{cll} \textbf{si } I=0 & \textbf{alors} & S\text{est indéfini} \\ \textbf{si S=0} & \textbf{alors} & T\text{est indéfini} \\ \textbf{si } B>G & \textbf{alors} & T=(360^\circ-T)/360^\circ \end{array}$$

Lorsque l'intensité est nulle, la couleur est alors forcément le noir. Sinon, quand la saturation est nulle, il s'agit d'un gris. L'intensité la plus forte est la couleur blanche. Enfin, le calcul de la teinte se faisant à l'aide de l'inverse du cosinus, il faut vérifier dans quel quadrant se situe le point de couleur P.

7. En anglais HSI : *Hue-Saturation-Intensity*.

8. En anglais HSV : *Hue-Saturation-Value*.

9. Les valeurs sont normalisées : comprises entre 0 et 1.

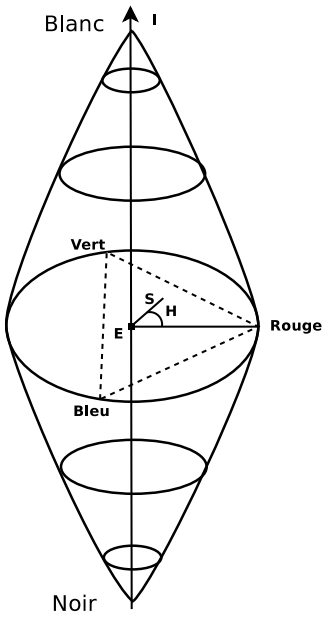


Figure 4.19. *L'espace des couleurs TSI : la mesure angulaire de la teinte se fait en référence à la couleur monochromatique rouge*

4.1.5.2. *L'espace TSV*

L'espace *TSV* est réputé être plus proche que l'espace TSI des critères développés initialement par les artistes peintres. Ces critères sont la *teinte*, la *nuance* et le *ton* :

$$\begin{cases} V & = & \max(R,G,B) \\ S & = & \frac{\max(R,G,B)-\min(R,G,B)}{\max(R,G,B)} \\ T & = & \arccos\left(\frac{(R-G)+(R-B)}{2\sqrt{(R-G)^2+(R-B)(G-B)}}\right) \end{cases}$$

La figure 4.20 illustre cet espace.

4.1.6. *Les espaces couleurs de la télévision*

Pour ne pas trop harmoniser les produits commerciaux, les normes couleurs de la télévision standard (SDTV) européenne et états-unienne diffèrent. Les Etats-Unis d'Amérique ont choisi le modèle *YIQ*, alors que l'Union Européenne a préféré le modèle *YUV*. Heureusement, le standard de la télévision haute définition (HDTV) est unique. Il s'agit du modèle *YCrCb*.

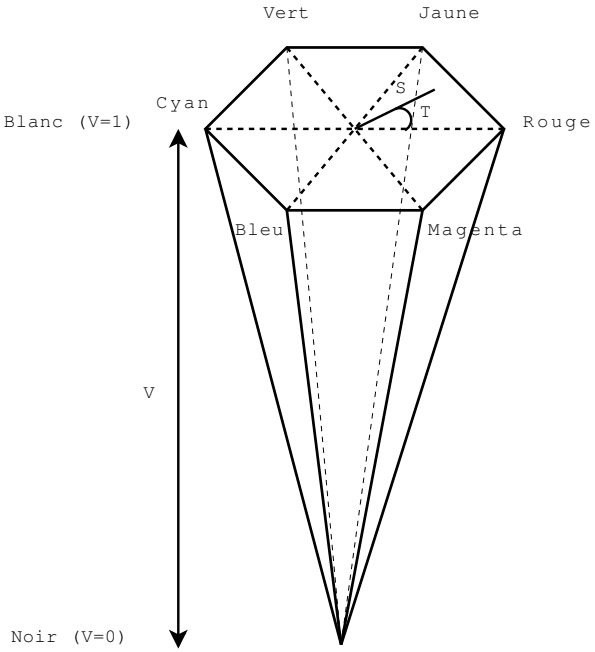


Figure 4.20. *L'espace de couleurs TSV*

Pour les espaces de représentation des couleurs de la SDTV, il était indispensable de séparer la luminosité de la chrominance afin d'assurer une continuité lorsque les téléviseurs couleurs sont apparus. Il fallait que les téléviseurs noir et blanc puissent continuer à visualiser les émissions émises en couleurs. Réciproquement, les téléviseurs couleurs devaient permettre de visualiser les émissions produites en noir et blanc.

Globalement, quel que soit l'espace choisi, la luminosité est donc toujours représentée par un axe indépendant, en forte relation avec l'axe Y de l'espace *XYZ* :

$$Y = 0,299R + 0,587G + 0,114B$$

La chrominance est alors décrite par les deux axes, C_1 et C_2, sous la forme suivante :

$$\begin{pmatrix} C_1 \\ C_2 \end{pmatrix} = \mathrm{A} \begin{pmatrix} R-Y \\ B-Y \end{pmatrix}$$

où $\mathrm{A} = (\alpha_{i,j})_{i,j}$ est la matrice 2×2 des coefficients $\alpha_{i,j}$. Elle dépend de l'espace choisi. Les transformations entre l'espace *RGB* et les espaces télévisuels sont linéaires, donc réversibles.

4.1.6.1. *L'espace YIQ*

L'espace *YIQ* est utilisé par le standard *National Television Standards Committee* (NTSC) de la télévision états-unienne. La chrominance est définie par :

$$\begin{pmatrix} I \\ Q \end{pmatrix} = \begin{pmatrix} 0,74 & -0,27 \\ 0,48 & 0,41 \end{pmatrix} \begin{pmatrix} R-Y \\ B-Y \end{pmatrix}$$

qui donne la transformation :

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0,299 & 0,587 & 0,114 \\ 0,596 & -0,274 & -0,322 \\ 0,212 & -0,523 & 0,311 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

4.1.6.2. *Les espaces YUV et YDrDb*

L'espace couleur *YUV* est celui choisi par le standard *Phase Alternation by Line* (PAL) de la SDTV européenne, hors France. Les axes de chrominance valent :

$$\begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} 0 & 0,493 \\ 0,877 & 0 \end{pmatrix} \begin{pmatrix} R-Y \\ B-Y \end{pmatrix}$$

qui donnent la transformation :

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0,299 & 0,587 & 0,114 \\ -0,147 & -0,289 & 0,436 \\ 0,615 & -0,515 & -0,1 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

La France a mis au point le standard *SEquentiel Couleur A Mémoire* (SECAM) qui utilise l'espace *YDrDb*. Les axes de chrominance valent :

$$\begin{pmatrix} \mathrm{Dr} \\ \mathrm{Db} \end{pmatrix} = \begin{pmatrix} -1,902 & 0 \\ 0 & 1,505 \end{pmatrix} \begin{pmatrix} R-Y \\ B-Y \end{pmatrix}$$

qui donnent la transformation :

$$\begin{pmatrix} Y \\ \mathrm{Dr} \\ \mathrm{Db} \end{pmatrix} = \begin{pmatrix} 0,299 & 0,587 & 0,114 \\ -1,333 & 1,116 & -0,217 \\ -0,45 & -0,883 & 1,333 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

4.1.6.3. *L'espace YCrCb*

Cet espace est le standard effectivement international. Il est utilisé par la télévision numérique (TN) et par le format JPEG2000 suivant un amendement de la partie 1. Il devrait devenir le standard couleur de la télévision haute définition (HDTV).

Alors que les standards précédents supposent une référence de blanc et d'autres paramètres comme la correction γ des écrans CRT, le standard *YCbCr* n'impose aucune condition. La matrice de passage de l'espace *RGB* à l'espace *YCbCr* vaut :

$$\begin{pmatrix} Y \\ \mathrm{Cr} \\ \mathrm{Cb} \end{pmatrix} = \begin{pmatrix} 0,299 & 0,587 & 0,114 \\ 0,5 & -0,419 & -0,081 \\ -0,169 & -0,331 & 0,5 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

4.2. Les propriétés de l'audio

Cette section présente les bases du codage audio numérique compressé. Ces bases utilisent les propriétés des sons, ainsi que celles du système auditif humain. Le paragraphe 4.2.1 présente la nature et la forme des sons. Le paragraphe 4.2.2 décrit les caractéristiques du système auditif humain. Ensuite, la section 4.2.3 expose les principes de la compression du signal audio utilisant les propriétés du son et du système auditif.

4.2.1. *Les sons*

Le son est une vibration macroscopique compressant ou dilatant les molécules d'air sous l'action d'un phénomène physique. Il est catalogué suivant la bande fréquentielle à laquelle il appartient. La *bande audible* pour l'humain se situe entre 20 Hz et 20 kHz. En deçà des 20 Hz, il s'agit d'*infrasons*. Les *ultrasons* sont situés dans la plage des 20 kHz à 1 GHz. Les *hypersons* occupent la bande des 1 GHz à 10 GHz. Dans cette section, seule la bande audible est étudiée.

La plupart du temps, un son est la combinaison complexe d'ondes sonores de différentes fréquences et de différentes formes : un bruit naturel, de la parole, de la musique, etc. Il peut être périodique ou apériodique.

Généralement, il est constitué d'une ou plusieurs *fondamentales*[10] et d'*harmoniques*. Les fondamentales sont souvent inférieures à 5 kHz. Les harmoniques se situent dans la bande fréquentielle de 5 kHz à 15 kHz.

Comme on l'a déjà constaté (voir chapitre 2), les amplitudes des harmoniques sont moindres que celles des fondamentales et vont en décroissant avec l'augmentation de la fréquence. Ainsi, l'énergie diminue avec l'augmentation de la fréquence.

10. Il peut y avoir plusieurs fondamentales comme dans le cas d'une musique où plusieurs instruments interviennent.

Cette propriété est exploitée par les systèmes de compression. Toutefois, les harmoniques jouent un rôle crucial dans la définition d'un son. Elles déterminent le *timbre* qui différencie une note jouée sur un piano de la même note jouée sur une guitare.

Le *niveau sonore* est mesuré en déciBell (dB[11]) :

– soit en fonction de la variation de pression acoustique :

$$L = 20 * \log_{10}\left(\frac{p}{p_{\mathrm{ref}}}\right)$$

où p est la *pression acoustique*, exprimée en Pascal (Pa). p_{ref} est le seuil minimal de perception (p_{ref} = 0,00002 Pa). On parle alors de *niveau de pression acoustique* ;

– soit en fonction de la variation de l'intensité acoustique :

$$L = 10 * \log_{10}\left(\frac{I}{I_{\mathrm{ref}}}\right)$$

où $I = \frac{P}{S}$ est la *intensité acoustique*, mesurée en puissance P, par mètre carré, S, ($W.m^{-2}$). I_{ref} est le seuil minimal de perception ($I_{\mathrm{ref}} = 10^{-12} W.m^{-2}$). On parle alors de *niveau d'intensité acoustique*.

Le niveau $0\,\mathrm{dB}$ n'est pas une mesure absolue, mais, correspond au seuil minimal d'audition chez l'humain. Ce seuil a été fixé expérimentalement et fournit les valeurs de p_{ref} et de I_{ref}.

Le tableau 4.2 fournit quelques exemples de sons et leurs intensités acoustiques.

Par ailleurs, un son n'apparaît ni ne disparaît instantanément. Tout son a une *enveloppe* qui le définit. Comme le montre la figure 4.21, un son commence par une *attaque*, suivie d'un léger *déclin* vers une zone *palier*. Il se termine par un temps d'*extinction*. La durée et l'amplitude de chacune de ces parties déterminent la forme de l'enveloppe, donc du son. Par exemple, une note sur une guitare a une attaque rapide et un relâchement lent alors que l'attaque d'un violon est plus lente et sa zone palier plus longue.

Cette définition du son permet de construire des modèles synthétiques de sons et en particulier d'instruments de musique. L'ensemble des synthétiseurs, matériels et logiciels, utilisent ces modèles. Typiquement, un modèle décrit une note musicale jouée par un instrument avec l'attaque et l'extinction voulues. Ce modèle est alors identifié par une séquence de codes indiquant les vitesses d'attaque, d'extinction, la note voulue, la durée du palier et bien d'autres paramètres. Une partition, créée à l'aide d'un synthétiseur, est donc une suite de codes.

11. Le dixième du Bell.

Intensité	Exemple de son
0dB	minimum audible
15 dB	des bruissements de feuilles
30 dB	des chuchotements
40 dB	une salle d'attente
60 dB	un ordinateur personnel de bureau à 0,6 mètres
65 dB	une voiture roulant à 60 km/h à 20 mètres
85 dB	un camion roulant à 50 km/h à 20 mètres
92 dB	une tondeuse à gazon motorisée à 1 mètre
95 dB	une rotative à journaux
103 dB	un métier à tisser
115 dB	un marteau pneumatique à 1 mètre
125 dB	un avion à réaction au décollage à 20 mètres
130 dB	*seuil de détérioration du système auditif*

Tableau 4.2. *L'intensité acoustique de divers sons (ref. http://fr.wikipedia.org/)*

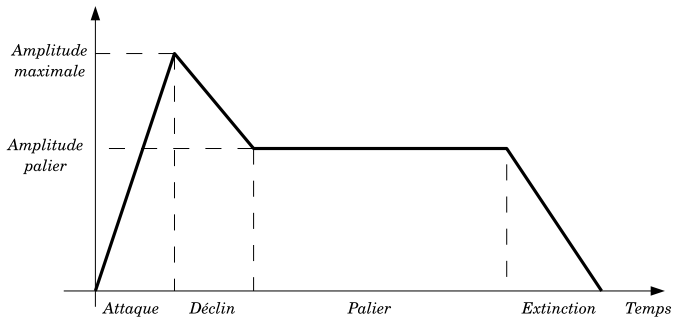


Figure 4.21. *Enveloppe d'un son*

A chaque code correspond un son qui est artificiel ou naturel :

1) le *son artificiel* est une composition de fonctions fréquentielles fondée sur l'étude, dans l'espace temps-fréquence, du son naturel correspondant ;

2) le *son naturel* provient d'un enregistrement audionumérique de l'instrument jouant la note voulue.

Les synthétiseurs et claviers actuels utilisent le format MIDI dont chaque code est composé de quatre à cinq champs (voir tableau 4.3) :

1) la *durée* (en millisecondes) codée par une technique à longueur variable ;

2) le *type d'événement* codé sur quatre bits (voir tableau 4.3) ;

3) le *canal* concerné codé sur quatre bits ;

4) le *premier paramètre* codé sur un octet ;

5) le *second paramètre* codé sur un octet.

Dans une première approche, les sites de Marc Chemillier et de l'IRCAM ainsi que le livre de Mrinal K. Mandal fournissent quelques compléments d'informations sur le format MIDI [MAN 03].

Type d'événement	Valeur (en hexadécimal)	Paramètre 1	Paramètre 2
Note On	`0x8`	numéro de note	vitess d'attaque
Note Off	`0x9`	numéro de note	vitesse d'extinction
Controller	`0xB`	numéro de contrôle	valeur de contrôle
Program Change	`0xC`	numéro de programme	aucune valeur
Pitch Bend	`0xE`	LSB de la hauteur (Pitch)	MSB de la hauteur

Tableau 4.3. *Exemples de types d'événements MIDI*

4.2.2. *Le système auditif humain*

Le système auditif humain est un processus complexe qui effectue une analyse temps-fréquence du signal sonore. La bande des fréquences étudiée est comprise entre 20 Hz et 20 kHz. Cette analyse identifie et localise dans le temps les fréquences du signal capté. On peut la comparer à la décomposition en ondelettes que l'on a vues au chapitre 2. Le principe d'incertitude de Heisenberg est donc toujours valable (voir section 2.6 page 39).

Pour le système auditif, il se traduit par un multifenêtrage dans le domaine temps-fréquence. Les plages fréquentielles des fenêtres ont été observées expérimentalement et sont données par le tableau 4.4.

Une fréquence f, produit un effet de *masquage* des fréquences voisines, si celles-ci sont d'intensités plus faibles. Le voisinage est défini par la fenêtre temps-fréquence de la fréquence f. La figure 4.22 montre la forme générale de la fonction de masquage dans le domaine fréquentiel.

EXEMPLE 4.1.– *Soit, par exemple, un signal sinusoïdal de fréquence 2 kHz et d'intensité* -6 *dB émis pendant 1 ms ; puis, pendant la milliseconde suivante, mixé avec un second signal sinusoïdal de fréquence 2,15 kHz. Le second signal n'est audible que si son énergie n'est pas inférieure de 20 dB à l'énergie du premier signal. L'expérimentation, si elle est inversée, fournit les mêmes résultats. Le mixage précède alors l'émission du signal pur. Lors de la première milliseconde, le signal à 2,15 kHz ne sera audible que si son énergie est supérieure à celle de l'autre sinusoïdale.*

Fréquence basse	Fréquence haute	Taille de l'intervalle	Fréquence basse	Fréquence haute	Taille de l'intervalle
20	100	80	2 000	2 320	320
100	200	100	2 320	2 700	380
200	300	100	2 700	3 150	450
300	400	100	3 150	3 700	550
400	510	110	3 700	4 400	700
510	630	120	4 400	5 300	900
630	770	140	5 300	6 400	1100
770	920	150	6 400	7 700	1300
920	1 080	160	7 700	9 500	1800
1 080	1 270	190	9 500	12 000	2500
1 270	1 480	210	12 000	15 500	3500
1 480	1 720	240	15 500	22 050	6550
1 720	2 000	280			

Tableau 4.4. *Les fenêtres du système auditif humain observées expérimentalement*

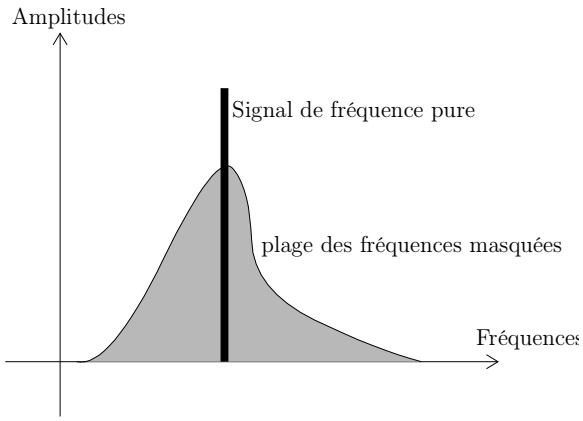


Figure 4.22. *Effet de masquage d'un signal de fréquence pure*

A première vue, le masquage temporel précédant le signal à son origine peut paraître surprenant. Toutefois, il ne faut pas oublier qu'il s'agit de l'interprétation des sons par l'oreille. Et non un phénomène physique des sons. Le fenêtrage du système auditif humain opère sur les fréquences *et* sur le temps. Ainsi, deux signaux dans la même plage temporelle vont interagir.

D'une manière générale, l'effet de masquage d'un signal faible par un signal fort ne peut avoir lieu que s'ils sont situés dans la même fenêtre temps-fréquence du système auditif. Par ailleurs, des études expérimentales sur la perception, en fonction de

l'intensité et de la fréquence du signal sonore,ont montré une sensibilité plus forte dans l'intervalle de fréquences $[1\,\mathrm{kHz}, 5\,\mathrm{kHz}]$ (voir figure 4.23). La courbe d'*isosonie* la plus basse est le *seuil d'audibilité* minimal.

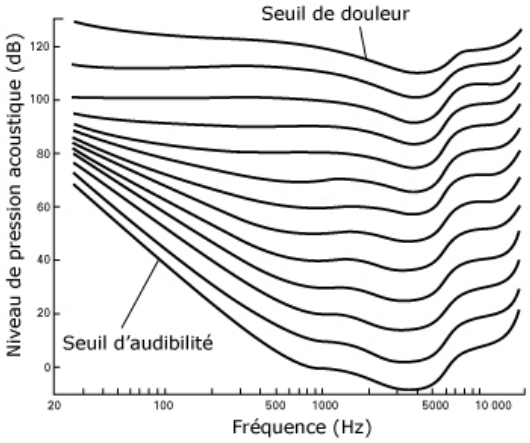


Figure 4.23. *Etude expérimentale de la sensibilité auditive : initialement, une sinusoïdale pure d'amplitude 20 dB et de fréquence 1 kHz est émise et perçue par un cobaye. Puis, la fréquence est modulée pour couvrir la plage* $[20\,\mathrm{Hz}, 20\,\mathrm{kHz}]$. *A chaque changement de fréquence, l'amplitude est adaptée pour que le cobaye conserve la même sensation d'intensité du signal. Une fois toute la plage des fréquences étudiée, on obtient la troisième courbe en partant du bas, appelée courbe d'isosonie. Le procédé est ensuite réitéré à l'identique avec le signal de* $1\,\mathrm{kHz}$*, mais, pour des amplitudes allant de 0 dB à 130 dB. Tous les tests ont été effectués avec des personnes d'une vingtaine d'années. Il résulte de cette étude qu'à amplitude constante, les basses fréquences* $([20\,\mathrm{Hz}, 1\,\mathrm{kHz}])$ *et les hautes fréquences* $([5\,\mathrm{kHz}, 20\,\mathrm{kHz}])$ *sont perçues plus faiblement.*

La combinaison des effets de masquage avec le seuil d'audibilité est illustrée en figure 4.24. La fréquence de plus forte intensité *déforme* le seuil d'audibilité au niveau pointillé de la courbe. Les deux fréquences voisines directes sont alors masquées. La fréquence la plus à droite sur la figure est d'intensité trop faible pour être audible dans tous les cas. Ces propriétés sont utilisées par les techniques de compression audio.

4.2.3. *Les bases de la compression audio*

Pour compresser au mieux le son, il faut distinguer ses diverses origines. Il peut être un *son naturel* comme la parole (conversation téléphonique) ou la musique enregistrée (un enregistrement analogique numérisé ou un enregistrement numérique) ou bien un *son synthétique* (généré par un synthétiseur). Cette section ne présente que les techniques de compression de la parole et du son naturel numérique.

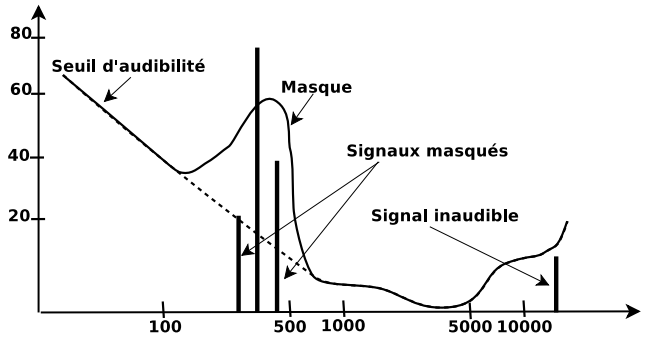


Figure 4.24. *Combinaison du masquage et du seuil d'audibilité*

4.2.3.1. *Compression de la parole*

La compression de la parole existe depuis longtemps puisque utilisée par le téléphone. Les techniques mises en œuvre sont relativement simples. Initialement, la bande fréquentielle des téléphones est assez étroite : $4\,\mathrm{kHz}$. L'échantillonnage est donc fait à une fréquence de $8\,\mathrm{kHz}$ avec l'utilisation d'un filtre passe-bas, si nécessaire, pour éliminer les hautes fréquences.

La compression doit être simple pour ne pas retarder la transmission de la parole. Autrement dit, la conversation téléphonique doit paraître instantanée. La quantification choisie est donc uniforme. Mais, si elle est appliquée directement sur les données échantillonnées, les basses fréquences, que sont les fondamentales, risquent d'être trop fortement quantifiées.

Il a été montré que la distribution d'un signal parole n'est pas uniforme – comme le requiert la quantification scalaire uniforme voir chapitre 3) – mais, qu'elle est centrée et de forme laplacienne. Afin de pouvoir utiliser la quantification scalaire uniforme, le signal doit donc être étiré jusqu'à obtenir une distribution quasi uniforme. La figure 4.25 montre le fonctionnement de la *compansion*[12].

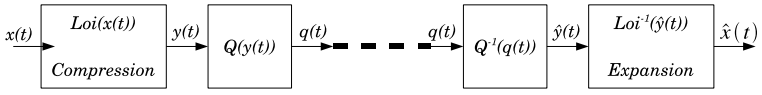


Figure 4.25. *Schéma général de la compansion*

12. *Compansion* est la contraction de *compression/expansion* qui est une traduction libre de *companding*, contraction anglaise de *compressing/expanding*.

Il existe deux lois standards réversibles qui permettent d'étirer le signal avant quantification et de le contracter après déquantification :

– la loi μ concerne principalement les Etats-Unis d'Amérique et le Japon :

$$y(t) = \text{sign}(x(t)) \frac{\log_2(1 + \mu|x(t)|}{\log_2(1 + \mu)}$$

La loi inverse vaut :

$$\hat{x}(t) = \text{sign}(\hat{y}(t)) \frac{(1 + \mu)^{|\hat{y}(t)|}}{\mu}$$

μ est un paramètre habituellement fixé à 255 (recommandation internationale) ;

– la loi A est utilisée en Europe (voir figure 4.26) :

$$y(t) = \begin{cases} \text{sign}(x(t)) \frac{A|x(t)|}{1+\log_2(A)} & \text{si } 0 \leqslant |x(t)| < \frac{1}{A} \\ \\ \text{sign}(x(t)) \frac{1+\log_2(A|x|)}{1+\log_2(A)} & \text{si } \frac{1}{A} \leqslant |x(t)| < 1 \end{cases}$$

La loi inverse vaut :

$$\hat{x}(t) = \begin{cases} \text{sign}(\hat{y}(t)) \frac{1}{A} |\hat{y}(t)|(1 + \log_2(A)) & \text{si } 0 \leqslant |\hat{y}(t)| < \frac{1}{1+\log_2(A)} \\ \\ \text{sign}(\hat{y}(t)) \frac{1}{A} e^{(|\hat{y}(t)|(1+\log_2(A))-1)} & \text{si } \frac{1}{1+\log_2(A)} \leqslant |\hat{y}(t)| < 1 \end{cases}$$

A est une constante habituellement fixée à 87,6 (recommandation internationale).

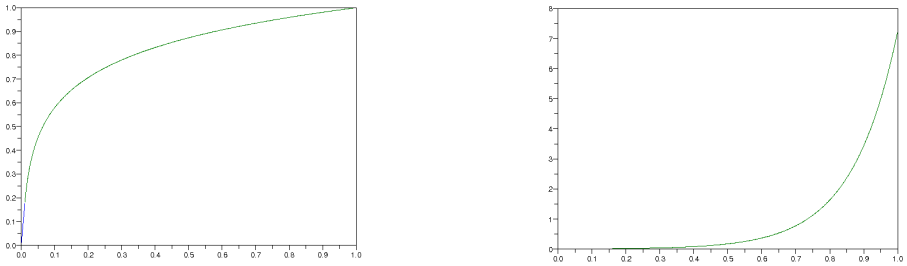


Figure 4.26. *A gauche : la loi A avec $A = 87,6$, à droite : son inverse*

4.2.3.2. *Les DPCM adaptatifs*

Les techniques de prédictions – vues en section 3.5 – sont intéressantes pour le codage du son qui, en général, n'est pas un signal aléatoire. Les échantillons sont fortement corrélés. La différence entre un échantillon et sa prédiction permet de réduire l'entropie de la source à coder. Mais, les techniques DPCM ne sont efficaces que si les

prédictions sont fidèles. Pour assurer une prédiction fidèle, il est possible d'adapter le processus au fur et à mesure que la source est codée. La figure 4.27 montre le schéma général du codage et du décodage prédictif adaptatif.

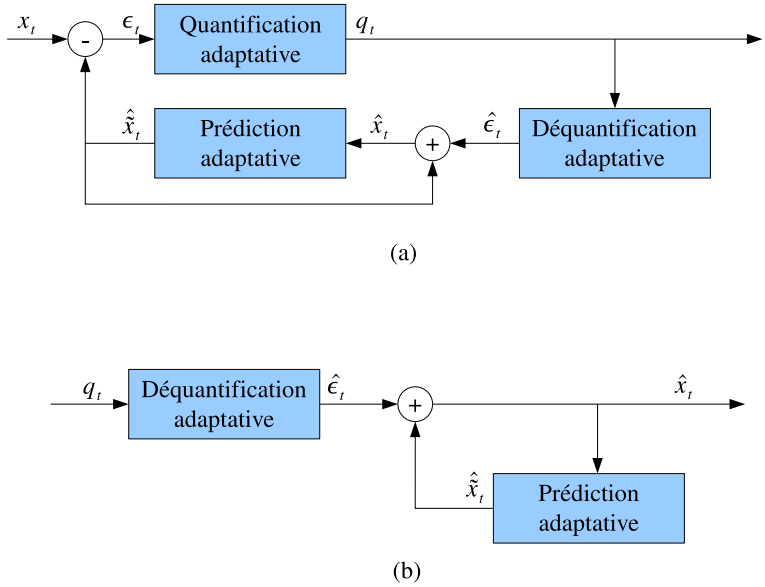


Figure 4.27. *Schéma général du ADPCM : (a) le codeur, (b) le décodeur*

Par exemple, des techniques de prédiction, prenant en compte plusieurs échantillons, estiment régulièrement les paramètres de prédiction. Ceci entraine un surcoût du débit, puisque, les paramètres sont à transmettre au décodeur. Mais, cette augmentation du débit est compensée par une compression accrue du signal.

Il est également possible d'adapter le processus de quantification aux valeurs des différences. Lorsque ces valeurs sont fortes, la quantification engendre moins d'erreurs si elle est plus fine dans la plage des hautes amplitudes. Inversement, une quantification précise dans la plage des amplitudes faibles minimise les erreurs pour des valeurs faibles.

EXEMPLE 4.2.– *Le format des DVD-Audio en est un exemple. Sa technique de quantification est scalaire. Trois seuils interviennent dans la construction d'un triplet de bits. Pour chaque seuil, si la valeur à coder lui est supérieure (resp. inférieure), le bit correspondant prend la valeur 1 (resp. 0). Le seuil associé au premier bit est fixé de manière adaptative à l'aide d'un jeu de deux tables de références. Les deuxième et troisième seuils sont, respectivement, la moitié et le quart du premier seuil.*

Le processus d'adaptation estime la valeur du premier seuil. Pour cela, une première table de références calcule les futurs indices d'une seconde table qui fournit la valeur du seuil. Les tables sont données par le tableau 4.5. La figure 4.28 illustre ce processus adaptatif.

```
QUANTIFICATION ADAPTATIVE DVD-AUDIO()
 1  b ← (0000)_2
 2  pour chaque échantillon x
 3  faire si x < 0
 4        alors b ← b + (1000)_2  // mise à 1 du 1er bit
 5              x ← −x  // c-à-d. du bit de signe
 6       si x > Qpas
 7        alors b ← b + (0100)_2
 8              x ← x − Qpas
 9       si x > Qpas/2
10        alors b ← b + (0010)_2
11              x ← x − Qpas/2
12       si x > Qpas/4
13        alors b ← b + (0001)_2
14              x ← x − Qpas/4
15       a ← 3 derniers bits de b
16       // estimation du pas de quantif. (voir tables en 4.5)
17       Qpas ← TableQ(a)
```

Table de référence n°1

3 bits	indice d'ajustement
000	−1
001	−1
010	−1
011	−1
100	2
101	4
110	6
111	8

Table de référence n°2

indice	pas de quantification
0	7
1	8
2	9
3	10
4	11
5	12
6	13
⋮	⋮

Tableau 4.5. *Tables de références*

4.2.3.3. *Prise en compte du système auditif humain*

Pour prendre en compte les caractéristiques du système auditif humain, la compression doit, en premier lieu, opérer une décomposition du signal audio en bandes de fréquences correspondant à celles du système auditif (voir tableau 4.4 page 151).

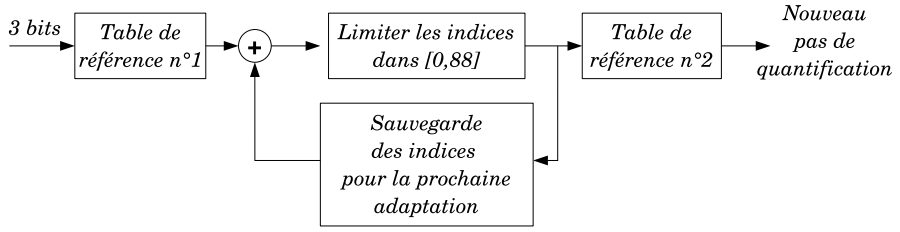


Figure 4.28. *Le processus d'adaptation du pas de quantification : à la réception des 3 bits, un indice d'ajustement est fourni par la première table. Celui-ci est ajouté à l'indice précédent. Une vérification est faite pour que le nouvel indice reste dans l'intervalle* $[0, 88]$. *Puis, cet indice est utilisé pour accéder à la nouvelle valeur du premier seuil dans la seconde table de référence.*

Typiquement, cette décomposition peut être effectuée en deux étapes. D'abord, les échantillons sont regroupés en blocs de façon à opérer la segmentation temporelle. Ensuite, chaque bloc est découpé en bandes de fréquences à l'aide d'un banc de filtres.

Par exemple, les formats MPEG utilisent un banc de 32 filtres de largeurs identiques (voir figure 4.29). A chaque filtre correspond un ensemble de paramètres modélisant le seuil d'audibilité et la fonction de masquage de la bande fréquentielle associée au filtre.

La figure 4.30 illustre le gain de codage lorsque le seuil d'audibilité et le masquage sont utilisés. Le nombre de bits, utilisés pour le codage de l'amplitude d'un échantillon, est proportionel au seuil d'audibilité (éventuellement modifié par la fonction de masquage).

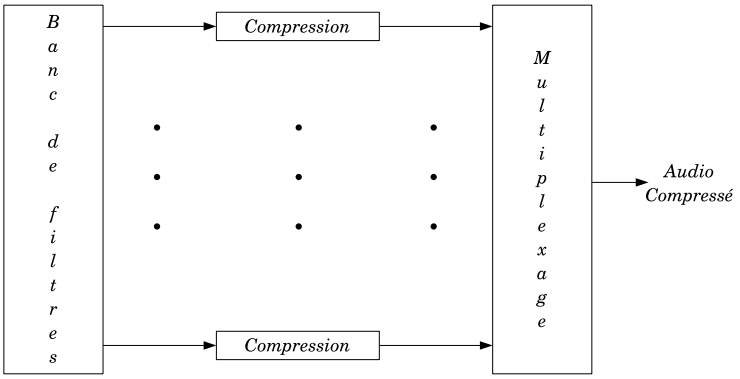


Figure 4.29. *Schéma général de compression à base de banc de filtres*

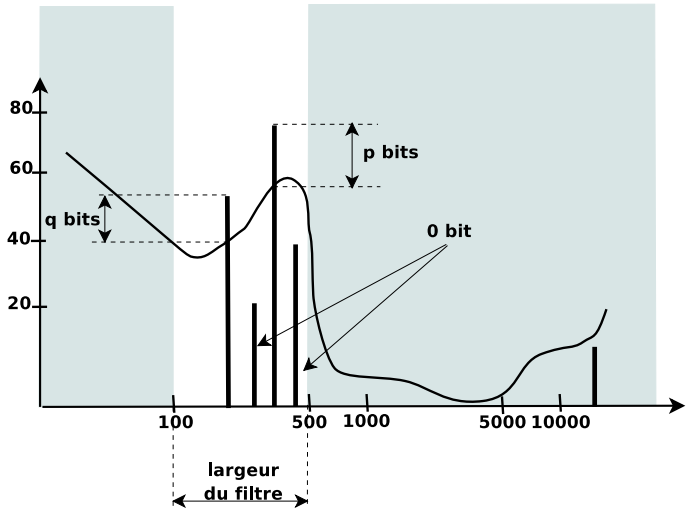


Figure 4.30. *Codage des deux échantillons au-dessus du seuil d'audibilité avec, respectivement, q et p bits. L'échantillon dominant – qui a généré la déformation du seuil d'audibilité – est codé avec un plus grand nombre de bits* $(p \geqslant q)$ *pour une meilleure définition*

A ceci vient s'ajouter des optimisations telles que :

– l'utilisation de la DCT ;

– une ADPCM d'ordre 2 (prédiction sur deux échantillons) ou plus ;

– une quantification non uniforme, etc.

Ces optimisations augmentent la qualité de l'enregistrement avec un gain supérieur en compression et permettent des débits faibles et variables. Nous reviendrons sur ces techniques dans les chapitres dédiés aux formats MPEG1 et MPEG2 (voir chapitres MPEG1 et MPEG2 du volume 2).

4.3. Synthèse

Ce chapitre vient de présenter le modèle visuel et le modèle auditif, couramment utilisés par l'informatique et l'audiovisuel. L'étude de ces sens montre que la perception humaine est *numérique*. Il reste donc au cerveau la lourde[13] tâche d'interpréter les signaux numériques perçus.

13. Lourde car difficile à faire effectuer par une machine !

L'œil est muni de capteurs qui filtrent les bandes spectrales rouge, verte et bleue. Mais, le travail de l'œil ne consiste pas uniquement à produire des échantillons numériques classés suivant leurs valeurs spectrales. Il est le seul capteur directement relié au cerveau et, à ce titre, il utilise toute une panoplie de neurones pour regrouper et organiser l'information visuelle.

Ainsi, l'œil fournit des informations de luminosité d'une part, et de chrominance, d'autre part. Les informations de luminosité sont prépondérantes dans toutes les étapes d'interprétation. Elles informent sur les mouvements, les contours et la texture des objets. La séparation entre luminosité et chrominance s'effectue par antagonisme des couleurs primaires dans l'espace RGB associé aux cônes L,M et S. L'espace résultant met en opposition le blanc avec le noir, le rouge avec le vert et le bleu avec le jaune. Ainsi, cet espace délivre un axe de luminosité et deux axes de chrominance.

Fort de cette constation, plusieurs modèles ont été développés avec des caratéristiques propres, adaptés à certains types de traitements. L'espace *XYZ* sert de référence à l'ensemble des autres espaces, car, il a été défini de telle sorte que toutes les couleurs soient représentées suivant le schéma de la synthèse additive.

Bien que communément utilisés dans le domaine de l'infographie, les espaces *RGB* sont des transformations linéaires de l'espace *XYZ* qui n'autorisent pas la représentation de toutes les couleurs. L'espace *RGB* défini par la CIE en 1931 est la référence internationale des espaces *RGB*. Ces espaces sont utilisés car ils définissent des primaires réellement existantes. *A contrario*, l'espace *XYZ* est fondé sur des primaires imaginaires afin d'assurer une réelle synthèse additive sur l'ensemble du spectre des couleurs.

L'espace $L^*a^*b^*$, quant à lui, est obtenu par une transformation non linéaire de l'espace *XYZ*. Il prend en compte les métamères. Ainsi, il permet une adéquation entre notre perception des différences de couleurs et la métrique euclidienne. Cet espace suppose, toutefois, que la valeur du blanc est *a priori* connue. Dans un même souci d'adéquation, les espaces TSI et TSV définissent un « *langage* » permettant de caractériser les couleurs.

Dans le cadre de la télévision et du multimédia, il existe plusieurs espaces couleurs. Tous reprennent la séparation, faite par l'espace XYZ, entre la chrominance et la luminosité. Ils indiquent la luminosité par l'axe Y avec la même valeur dans tous ces espaces. Les espaces *YIQ*, *YUV* et *YDrDb* de la télévision standard et l'espace *YCrCb* du multimédia et de la télévision haute définition (HDTV) sont décrits.

Notre deuxième sens intervenant en audiovisuel est l'ouïe. Les sons perçus peuvent être naturels ou synthétiques.

Les sons synthétiques reposent généralement sur l'utilisation d'échantillons préenregistrés de notes, jouées par des instruments qu'une suite de codes permet d'assembler et de moduler. Dans ce cadre, la compression est de type compression de texte bien que les codes générés soient déjà optimisés pour être joués en temps réel.

Les sons naturels peuvent être plus ou moins complexes, plus ou moins riches en fréquences : la parole, la musique, etc.

Pour la parole, la technique de compansion est utilisée depuis longtemps pour la conversation téléphonique. Avec l'arrivée du signal numérique, d'autres techniques ont été définies, notamment par les derniers formats MPEG.

Pour les sons plus complexes, les techniques adaptatives du DPCM apportent une solution efficace pour un coût restant faible. Mais, pour une qualité correcte, voire haute, du son, les propriétés du système auditif humain sont prises en compte. Ainsi, un banc de filtres effectue la décomposition dans l'espace temps-fréquence du son. A chaque bande de fréquences est associée un modèle du seuil d'audibilité et de masquage. Les amplitudes des échantillons sont alors codées proportionnellement au seuil d'audibilité (éventuellement modifié par un masquage). Ces techniques, ainsi que les optimisations correspondantes, se retrouvent à nouveau dans les formats MPEG et seront donc vus dans les chapitres dédiés.

Avec ce chapitre, on conclut la première partie de ce livre. Celle-ci fournit une description générale des concepts de base et une introduction aux théories associées. Les transformées sont vues en premier, car elles sont en relation avec pratiquement tous les autres concepts. Ensuite, sont abordés, dans l'ordre, l'échantillonnage, la quantification et le codage. La quantification et le codage repose sur la théorie de l'information afin d'en comprendre les fondements. En dernier, les modèles visuel et auditif humains sont décrits.

Les parties suivantes vont utiliser et développer ces concepts et techniques pour faire de la compression d'images (voir partie image du volume 2) et de la compression audiovidéo (voir partie video du volume 2).

Annexe A

Compléments : les transformées

A.1. Temps réel

Dans le domaine de la compression, certain codeur peuvent compresser les données à la volée. Un codeur de ce genre n'attend pas d'avoir codé l'ensemble de la source pour l'envoyer. Il construit des segments de données compressées qu'il envoie régulièrement au décodeur.

Le codeur doit alors s'assurer de l'envoie régulier de données pour que le décodeur n'ait pas à les attendre. A l'inverse, il doit aussi s'assurer que la mémoire en entrée du décodeur n'est pas pleine avant de lui envoyer de nouvelles données.

Par exemple, lors de la location d'un film sur un Internet (on parle de *film à la demande* : VOD), l'application de visualisation doit toujours avoir des images à afficher (données décompressées). Si le codeur n'envoie pas de données assez régulièrement (*underflow*), la fluidité de la visualisation s'en ressentira. Au contraire, si le codeur envoie des segments audiovisuels alors que la mémoire en entrée du décodeur est pleine (*overflow*), ceux-ci seront perdus.

Un codeur effectuant un codage à la volée en évitant ces deux situations critiques est dit *temps réel*.

A.2. Corrélation

Un signal est décrit par un certain nombre de dimensions comme le temps, la fréquence, la position spatiale 2D ou 3D, etc.

Dans le cas d'un signal sonore sa dimension usuelle est le temps. Un échantillon de ce signal est une amplitude enregistrée à un instant précis. Il est habituellement en relation avec les échantillons qui l'ont précédée dans le temps. On parle alors de *corrélation* entre les échantillons.

Par exemple, en musique, chaque note correspond à un intervalle dans le temps. Cet intervalle est décrit par plusieurs échantillons qui ont en commun la fréquence principale de la note.

Si, en revanche, les échantillons du signal ne sont liés par aucune relation, on dit qu'ils sont *indépendants*.

Décorréler un signal consiste à le décrire à partir de caractéristiques indépendantes.

Si l'on considère un rectangle dans un espace 2D, trois attributs suffisent à le caractériser : son centre (défini par l'intersection de ses diagonales), sa largeur et sa longueur. Ces caractéristiques sont indépendantes les unes des autres. Une homothétie appliquée au rectangle change sa longueur et sa largeur mais pas son centre. On peut aussi déplacer le rectangle en modifiant uniquement son centre. Ces trois attributs sont indépendants car la modification d'un n'implique pas le changement des deux autres.

A.3. Contexte

Le *contexte* d'un échantillon est défini par le voisinage de celui-ci.

Si l'échantillon provient d'un signal temporel (1D), le contexte est défini par les échantillons voisins dans le temps. Ce voisinage porte sur les échantillons passés mais également sur quelques échantillons futurs.

Si l'échantillon provient d'une image, il s'agit d'un *pixel*. Le contexte est alors défini par les pixels qui lui sont proches. La forme du voisinage dépendant souvent du capteur qui saisit l'image.

Si l'image est perçue par un appareil photographique numérique (APN). Tous ses pixels sont valués au même instant. Le voisinage est alors un disque de centre l'échantillon observé. Le rayon du disque définit la taille du voisinage.

Si l'image est lue par un scanner, elle est enregistrée ligne par ligne. Le contexte est alors défini par les pixels se situant sur les lignes déjà parcourues par le scanner et par les pixels de la ligne en cours de lecture qui précèdent l'échantillon.

A.4. Fonctions et signaux périodiques

Un signal *périodique* est un signal dont la valeur pour une variable donnée se répète quand on ajoute une quantité *constante* à la variable. cette quantité est appelée période :

$$s(x) = s(x+p) \text{ où } p \text{ est la période de valeur constante.}$$

Son contraire est un signal *apériodique*.

A.5. Fonctions et signaux discrets

Une fonction continue possède des variables à *valeurs continues*. Cette définition est volontairement grossière, car une définition plus rigoureuse mathématiquement sort du cadre de ce livre [GAS 00].

Une fonction discrète possède des variables à *valeurs entières*. Mais, son résultat est quelconque. C'est le cas des signaux numériques.

Par exemples, un signal musical numérique est constitué d'échantillons régulièrement espacés dans le temps dont les amplitudes sont réelles; une image au format RGB est composée de pixels (indicés par des couples d'entiers) dont les valeurs sont des triplets d'entiers.

A.6. $\mathbb{C}$: les nombres complexes

Une variable complexe v, possède une partie réelle a, et une partie imaginaire b :

$$v = a + ib$$

La constante i est l'*unité imaginaire pure* ayant la propriété suivante :

$$i^2 = -1$$

Le complexe v peut également être décrit par son module, $\|v\|$, et sa phase, $\varphi(v)$ (voir figure A.1) :

$$\begin{array}{rcl} \|v\| & = & \sqrt{a^2+b^2} \\ \varphi(v) & = & \arctan\left(\frac{b}{a}\right) \end{array}$$

Le conjugué de v, noté $v^\star$, vaut :

$$v^\star = a - ib$$

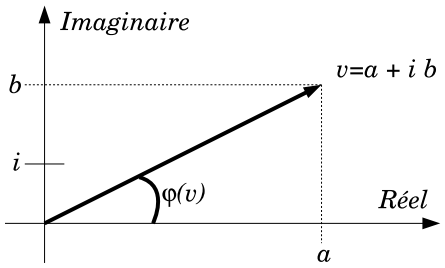


Figure A.1. *Représentation des nombres complexes*

A.6.1. *Fonctions complexes*

Soit une fonction complexe :

$$g(x) = a(x) + ib(x)$$

où $a(x)$ et $b(x)$ sont des fonctions réelles.

Le conjugué, noté $g^\star$, de g est défini par :

$$g^\star(x) = a(x) - ib(x)$$

Si la fonction g est réelle, son conjugué est la fonction elle-même :

$$g^\star(x) = g(x)$$

A.7. Projections et produit scalaire

DÉFINITION A.1.– *Le produit scalaire entre les vecteurs* $\boldsymbol{u} = (u_i)$ *et* $\boldsymbol{v} = (v_i)$ *s'écrit :*

$$< \boldsymbol{u}, \boldsymbol{v} >= \sum_i u_i * v_i$$

EXEMPLE A.1.– *On peut vérifier les propriétés du produit scalaire appliqué aux vecteurs de la base orthonormée de l'espace tridimensionnel :*

$$\boldsymbol{e}_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \boldsymbol{e}_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \boldsymbol{e}_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Les vecteurs $\boldsymbol{e}_0$ et $\boldsymbol{e}_1$ étant orthogonaux, leur produit scalaire est nul :

$$< \boldsymbol{e}_0, \boldsymbol{e}_1 >= 1 * 0 + 0 * 1 + 0 * 0 = 0$$

On en déduit une écriture de l'orthogonalité de la base $(\boldsymbol{e}_i)$:

$$< \boldsymbol{e}_i, \boldsymbol{e}_j >= \begin{cases} 1 & \textit{si } i = j \\ 0 & \textit{sinon} \end{cases}$$

PROPOSITION A.1.– *Le produit scalaire permet de calculer les projections sur les vecteurs de la base.*

EXEMPLE A.2.– *Soit le vecteur $u = (3.15, 5.89, 0.18)$. Son projeté sur le premier axe vaut :*

$$< \boldsymbol{u}, \boldsymbol{e}_0 >= 1 * 3.15 + 0 * 5.89 + 0 * 0.18 = 3.15$$

DÉFINITION A.2.– *Le produit scalaire de deux vecteurs complexes $\boldsymbol{u}$ et $\boldsymbol{v}$, s'écrit :*

$$< \boldsymbol{u}, \boldsymbol{v} >= \sum_i u_i * v_i^\star$$

avec $v_i^\star$ le conjugué de v_i.

Cette définition généralise la première aux nombres complexes, car, si v est réel, son conjugué est lui-même : $v^\star = v$.

A.7.1. *Produit scalaire de fonctions*

Le produit scalaire peut également être appliqué aux fonctions continues sous certaines conditions [GAS 00]. La sommation se fait par intégration.

Soient deux fonctions f et g. Leur produit scalaire s'écrit :

$$< f, g >= \int_I f(x) g^\star(x) \mathrm{d}x$$

où $g^\star$ est le conjugué de la fonction g.

A.8. Bases d'exponentielles complexes

Une exponentielle complexe est définie comme suit :

$$e^{i\theta} = \cos(\theta) + i\sin(\theta)$$

Les fonctions cosinus et sinus se réécrivent avec les exponentielles complexes :

$$\begin{array}{rcl} \cos(\theta) & = & \frac{1}{2}\left(e^{i\theta} + e^{-i\theta}\right) \\ \sin(\theta) & = & -\frac{i}{2}\left(e^{i\theta} - e^{-i\theta}\right). \end{array}$$

Cette réécriture permet d'introduire la forme bilatérale des séries de Fourier (voir paragraphe 2.2.2 page 25). Les coefficients $a(nF)$ et $b(nF)$, des séries de Fourier sont alors en relation avec le coefficient $S(nF)$, de la forme bilatérale :

$$\begin{array}{rcl} S(nF) & = & \frac{1}{2}\left(a(nF) - ib(nF)\right) \\ & = & \frac{1}{T}\int_0^T s(t)e^{-i2\pi nFt}\mathrm{d}t \end{array}$$

En particulier, on retrouve :

$$S(0) = \frac{a(0)}{2}$$

A.8.1. *Orthogonalité des exponentielles complexes*

La famille $\{e^{i2\pi nFt}\}$forme une *base orthogonale* [GAS 00, MAL 98] :

$$< e^{i2\pi n_1 Ft}, e^{i2\pi n_2 Ft} >= \delta(n_1 - n_2) = \left\{ \begin{array}{ll} 1 & \textbf{si } n_1 = n_2 \\ 0 & \textbf{sinon} \end{array} \right.$$

où δ est appelée l'*impulsion de Dirac* qui vaut 1 en 0 et 0 partout ailleurs. Sa définition est donnée en annexe B.3 (page 174).

Le produit scalaire appliqué aux exponentielles complexes s'écrit :

$$\begin{array}{rcl} < e^{i2\pi n_1 Ft}, e^{i2\pi n_2 Ft} > & = & \int_{-\pi}^{+\pi} e^{i2\pi n_1 Ft} e^{-i2\pi n_2 Ft}\mathrm{d}t \\ & = & \int_{-\pi}^{+\pi} e^{i2\pi(n_1-n_2)Ft}\mathrm{d}t \\ & = & \left\{ \begin{array}{ll} \int_{-\pi}^{+\pi} e^{i0}\mathrm{d}t = 1 & \text{si } n_1 = n_2 \\ \int_{-\pi}^{+\pi} e^{i2\pi nFt}\mathrm{d}t = 0 & \text{sinon}(n = n_1 - n_2) \end{array} \right. \end{array}$$

A.9. Propriétés de la série de Fourier

THÉORÈME A.1.– *Le développement en série de Fourier est linéaire :*

$$\alpha x(t) + \beta y(t) \overset{\mathfrak{F}}{\Longleftrightarrow} \alpha X(f) + \beta Y(f)$$

avec α et β des constantes réelles.

THÉORÈME A.2.– *Il possède la propriété de parité définie par le tableau qui suit :*

signal $s(t)$	***spectre fréquentiel*** $S(f)$
$s(t)$ *est réel et pair*	$b_n = 0, \forall n \in \mathbb{N}$; $S(f)$ *est réel et pair*
$s(t)$ *est réel et impair*	$a_n = 0, \forall n \in \mathbb{N}$; $S(f)$ *est imaginaire et impair*
$s(t)$ *est réel et quelconque*	$S(f)$ *est complexe avec une partie réelle paire et une partie imaginaire impaire*

A.10. Les propriétés de la transformée de Fourier

THÉORÈME A.3.– *Les propriétés de la transformée de Fourier sont :*

– la linéarité :

$$ax(t) + by(t) \overset{\mathfrak{F}}{\Longleftrightarrow} a\hat{X}(f) + b\hat{Y}(f)$$

– l'homothétie :

$$x(at) \overset{\mathfrak{F}}{\Longleftrightarrow} \frac{1}{|a|}\hat{X}\left(\frac{f}{a}\right)$$

avec $a \in \mathbb{R}$;

– la translation :

$$x(t - t_0) \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{X}(f)e^{i2\pi t_0 f}$$

avec $t_0 \in \mathbb{R}$;

$$x(t)e^{i2\pi f_0 t} \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{X}(f - f_0)$$

avec $f_0 \in \mathbb{R}$;

– la dérivation :

$$\frac{\mathrm{d}^n}{\mathrm{d}t^n}(x(t)) \overset{\mathfrak{F}}{\Longleftrightarrow} (i2\pi f)^n \hat{X}(f)$$

– si $x(t)$ est complexe alors son conjugué $x^\star$, est en relation avec la représentation spectrale $\hat{X}(f)$:

$$x^\star(t) \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{X}(-f)$$

– *la parité :*

signal $x(t)$	***spectre fréquentiel*** $\hat{X}(f)$
$x(t)$ *est réel et pair*	$\hat{X}(f)$ *est réel et pair*
$x(t)$ *est réel et impair*	$\hat{X}(f)$ *est imaginaire et impair*
$x(t)$ *est réel et quelconque*	$\hat{X}(f)$ *est complexe avec une partie réelle paire et une partie imaginaire impaire*
$x(t)$ *est imaginaire et quelconque*	$\hat{X}(f)$ *est complexe avec une partie réelle impaire et une partie imaginaire paire*
$x(t)$ *est imaginaire et pair*	$\hat{X}(f)$ *est imaginaire et pair*
$x(t)$ *est imaginaire et impair*	$\hat{X}(f)$ *est réel et impair*
$x(t)$ *est complexe et pair*	$\hat{X}(f)$ *est complexe et pair*
$x(t)$ *est complexe et impair*	$\hat{X}(f)$ *est complexe et impair*

A.11. Convolution

Le domaine du filtrage linéaire et de la *convolution* est bien trop vaste pour être étudié dans ce livre. Cette section va limiter l'étude aux propriétés de la convolution, car elle intervient dans certains outils du multimédia.

La convolution s'apparente au calcul d'une moyenne pondérée. Un signal, $x(t)$, est convolué par un *filtre* $h(t)$, suivant l'équation suivante :

$$y(t) = x(t) \otimes h(t) = \int_{-\infty}^{+\infty} x(t)h(\tau - t)\mathrm{d}\tau$$

Le filtre est centré à l'origine et décroît fortement. Un exemple typique de filtre est la *gaussienne*. Celle-ci a pour effet de lisser le signal $x(t)$, c'est-à-dire, de minimiser les brusques changements apparaissant dans le signal x.

La *version discrète* de la convolution du signal discret $x[n]$, par le filtre discret, $h[n]$, s'écrit :

$$y[n] = x[n] \otimes h[n] = \sum_{m=-\infty}^{+\infty} x[n]h[m-n] \tag{A.1}$$

REMARQUE A.1.– *Les crochets* [*et*] *sont utilisés pour distinguer les signaux discrets des signaux continus.*

Les filtres utilisés dans ce livre sont à *support fini*. Autrement dit, le filtre discret $h[n]$, est nul en dehors d'un intervalle $I = [a, b] \neq \emptyset$. Ceci s'implifie l'équation (A.1) :

$$y[n] = \textstyle\sum_{m=a}^{b} x[n]h[m-n]$$

EXEMPLE A.3.– *Soit le filtre* $h = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$ *et le signal* $x = [1, 1, 5, 2, 1, 1, 7, 7, 7, 1, 7]$, *avec* $x[0] = 1$. *La convolution du signal par le filtre est expliquée pas à pas :*

$$\begin{aligned}
y[1] &= 1 * \tfrac{1}{3} + 1 * \tfrac{1}{3} + 5 * \tfrac{1}{3} &= \tfrac{7}{3} \\
y[2] &= 1 * \tfrac{1}{3} + 5 * \tfrac{1}{3} + 2 * \tfrac{1}{3} &= \tfrac{8}{3} \\
y[3] &= 5 * \tfrac{1}{3} + 2 * \tfrac{1}{3} + 1 * \tfrac{1}{3} &= \tfrac{8}{3} \\
y[4] &= 2 * \tfrac{1}{3} + 1 * \tfrac{1}{3} + 1 * \tfrac{1}{3} &= \tfrac{4}{3} \\
y[5] &= 1 * \tfrac{1}{3} + 1 * \tfrac{1}{3} + 7 * \tfrac{1}{3} &= \tfrac{9}{3} \\
y[6] &= 1 * \tfrac{1}{3} + 7 * \tfrac{1}{3} + 7 * \tfrac{1}{3} &= \tfrac{15}{3} \\
y[7] &= 7 * \tfrac{1}{3} + 7 * \tfrac{1}{3} + 7 * \tfrac{1}{3} &= \tfrac{21}{3} \\
y[8] &= 7 * \tfrac{1}{3} + 7 * \tfrac{1}{3} + 1 * \tfrac{1}{3} &= \tfrac{15}{3} \\
y[9] &= 7 * \tfrac{1}{3} + 1 * \tfrac{1}{3} + 7 * \tfrac{1}{3} &= \tfrac{15}{3}
\end{aligned}$$

Chaque valeur de y *est la moyenne de trois valeurs de* x. h *est un filtre moyenneur. La figure A.2 montre le signal original, le filtre et le résultat de la convolution. Le signal* y *lisse les valeurs extêmes locales comme celle qui est en position* $n = -3$. *Ceci est dû au type (moyenneur) du filtre utilisé. D'autres filtres feront ressortir les changements abruptes du signal [COT 02, VAN 03].*

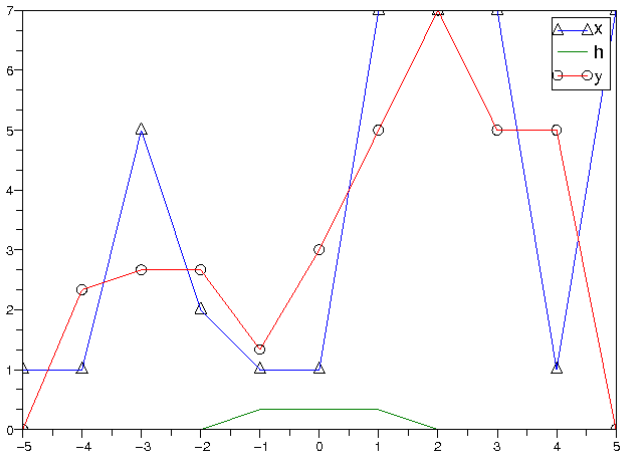


Figure A.2. *Résultat de la convolution du signal* x *par le filtre* h

A.12. Implémentation de la transformée de Fourier 2D

Lors de l'implémentation de la transformée de Fourier 2D, la propriété de séparabilité peut être utilisée :

$$\begin{aligned}\hat{S}[n,m] &= \frac{1}{NM}\sum_{k=0}^{N-1}\sum_{j=0}^{M-1} s[k,j]e^{-i2\pi\left(n\frac{k}{N}+m\frac{j}{M}\right)} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\left(\frac{1}{M}\sum_{j=0}^{N-1} s[k,j]e^{-i2\pi m\frac{j}{M}}\right)e^{-i2\pi n\frac{k}{N}} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\hat{S}[k,m]e^{-i2\pi n\frac{k}{N}}\end{aligned}$$

avec :

$$\hat{S}[k,m] = \frac{1}{M}\sum_{j=0}^{N-1} s[k,j]e^{-i2\pi m\frac{j}{M}}$$

Ceci permet d'écrire un algorithme qui opère deux transformées de Fourier 1D de suite : une première transformée selon les lignes, puis une seconde selon les colonnes, sur le résultat de la première transformée 1D.

A.13. L'analyse multirésolution

Pour effectuer une analyse multirésolution à l'aide de la fonction d'échelle φ, il faut fixer certaines propriétés [PES 01] :

1) le sous-espace vectoriel, V_i, de niveau de résolution i est inclus dans le sous-espace vectoriel, V_{i-1}, de niveau de résolution $i-1$;

2) une fonction $s(t)$ appartient au sous-espace vectoriel V_{i-1} si et seulement si sa dilatée $s(t/2)$ appartient au sous-espace vectoriel V_i ;

3) l'ensemble $\{\varphi(t-m), m \in \mathbb{Z}\}$, défini à partir de la fonction d'échelle φ, forme une base orthogonale de V_0.

Le point 3 signifie, en particulier, que la fonction d'échelle du niveau 1, $\frac{1}{\sqrt{2}}\varphi\left(\frac{t}{2}\right)$, est une combinaison linéaire des fonctions d'échelle du niveau 0, $\varphi(t-m)$:

$$\frac{1}{\sqrt{2}}\varphi\left(\frac{t}{2}\right) = \sum_{m\in\mathbb{Z}} h[m]\varphi(t-m)$$

Cette équation liant la fonction d'échelle dilatée aux fonctions d'échelles déplacées est appelée *équation aux deux échelles*.

Grâce à l'orthogonalité de la base, les coefficients s'obtiennent par produit scalaire :

$$h[m] =< \frac{1}{\sqrt{2}}\varphi(\frac{t}{2}), \varphi(t-m) > \tag{A.2}$$

En généralisant, on a la relation entre les niveaux $j+1$ et j :

$$2^{-\frac{j+1}{2}}\varphi\left(2^{-(j+1)}t-k\right) = \sum_{m\in\mathbb{Z}} h[m-2k]2^{-\frac{j}{2}}\varphi\left(2^{-j}t-m\right)\forall k\in\mathbb{Z} \tag{A.3}$$

Le sous-espace vectoriel complémentaire au sous-espace des fonctions d'échelles de niveau 1 est le sous-espace vectoriel des fonctions d'ondelettes de niveau 1 (voir figure A.3). De ce fait, on a la même équation aux deux échelles entre les fonctions d'ondelettes et les fonctions d'échelles:

$$2^{-\frac{j+1}{2}}\psi\left(2^{-(j+1)}t-k\right) = \sum_{m\in\mathbb{Z}} g[m-2k]2^{-\frac{j}{2}}\varphi\left(2^{-j}t-m\right)\forall k\in\mathbb{Z} \tag{A.4}$$

Il est montré que la suite $h[m]$ se comporte comme un filtre passe-bas et la suite $g[m]$ comme un filtre passe-haut [PES 01].

NOTE A.1.– *Puisque les signaux étudiés sont discrets et que les fonctions d'échelles et d'ondelettes sont des bandes passantes, leurs filtres, h et g, sont de longueurs finies, L; c'est-à-dire avec un nombre fini de valeurs non nulles. Ils sont appelés filtres à réponse impulsionnelle finie (RIF).*

THÉORÈME A.4.– *Dans les équations (A.3) et (A.4), les fonctions d'échelle et d'ondelette $\varphi\left(2^{-(j+1)}t-k\right)$ et $\psi\left(2^{-(j+1)}t-k\right)$, du niveau $j+1$, sont obtenues par convolution de la fonction d'échelle $\varphi(2^{-j}t-m)$, avec les filtres $h[m-2k]$ et $g[m-2k]$ (accompagnée d'un sous-échantillonnage).*

PRINCIPE D'ANALYSE A.0.– *Les coefficients des fonctions d'échelle, $\lambda_{j+1}[k]$, et d'ondelette, $\gamma_{j+1}[k]$, du niveau $j+1$, s'obtiennent à partir des coefficients d'échelle, $\lambda_j[k]$, du niveau j :*

$$\begin{aligned}\lambda_{j+1}[k] &= \textstyle\sum_{l=0}^{L-1} h[l-2k]\lambda_j[l]\\ \gamma_{j+1}[k] &= \textstyle\sum_{l=0}^{L-1} g[l-2k]\lambda_j[l]\end{aligned}$$

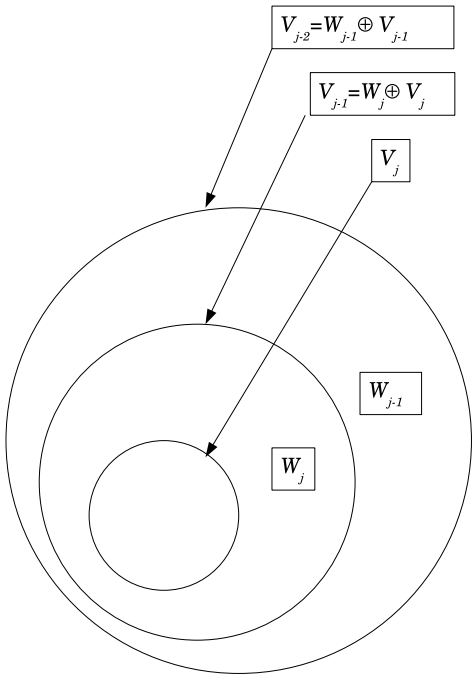


Figure A.3. *V_j est le sous-espace des fonctions d'échelle j. W_j est le sous-espace des fonctions d'ondelettes d'échelle j*

PRINCIPE DE SYNTHÈSE A.0.– *Les coefficients d'échelle du niveau j s'obtiennent à partir des coefficients d'échelle, λ_{j+1}, et des coefficients d'ondelette, γ_{j+1}, du niveau j :*

$$\lambda_j[k] = \sum_{l=0}^{L-1} h[k-2l]\lambda_{j+1}[l] + \sum_{l=0}^{L-1} g[k-2l]\gamma_{j+1}[l]$$

Annexe B

Compléments : numérisation et codage

B.1. Energie et puissance moyenne

L'énergie est une mesure cumulée de la puissance instantanée. Par exemple, la puissance instantanée d'une résistance électrique R, vaut :

$$p(t) = u(t)i(t) = Ri^2(t)$$

avec $u(t)$ et $i(t)$ les mesures instantanées respectives de la tension et de l'intensité du courant électrique traversant la résistance. L'énergie dissipée par cette résistance dans l'intervalle de temps $[t_1, t_2]$:

$$W(t_1, t_2) = \int_{t_1}^{t_2} p(t)\mathrm{d}t = R\int_{t_1}^{t_2} i^2(t)$$

se mesure en joules (J). La puissance moyenne exprime la mesure moyenne de cette énergie pendant cet intervalle de temps :

$$P(t_1, t_2) = \frac{W(t_1, t_2)}{t_2 - t_1}$$

Elle se mesure en watts (W).

Pour un signal $s(t)$, on a l'analogie suivante :

$$W_s(t_1, t_2) = \int_{t_1}^{t_2} s^2(t)\mathrm{d}t \tag{B.1}$$

$$P_s(t_1, t_2) = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} s^2(t)\mathrm{d}t \tag{B.2}$$

Sur l'intervalle de temps infini, les équations (B.1) et (B.2) s'écrivent :

$$\begin{aligned} W_s &= \int_{-\infty}^{+\infty} s^2(t)\mathrm{d}t \\ P_s &= \lim_{T\to+\infty} \frac{1}{T} \int_{-T/2}^{+T/2} s^2(t)\mathrm{d}t \end{aligned}$$

Et pour un signal périodique, de période T_0, la puissance moyenne est définie sur la période :

$$P_s = \frac{1}{T_0} \int_{-T_0/2}^{+T_0/2} s^2(t)\mathrm{d}t$$

B.2. Le signal rectangle

Le signal rectangulaire est défini par :

$$\Pi_\tau(t) = \begin{cases} A & \text{si } |t| \leqslant \tau \\ 0 & \text{sinon} \end{cases}$$

Sa transformée de Fourier vaut :

$$\begin{aligned} \hat{\Pi}_\tau(f) &= \int_{-\infty}^{+\infty} \Pi_\tau(t) e^{-i2\pi\,\mathrm{ft}}\mathrm{d}t \\ &= \int_{-\tau}^{+\tau} A e^{-i2\pi\,\mathrm{ft}}\mathrm{d}t \\ &= A\frac{\sin(2\pi f\tau)}{\pi f} \\ &= 2A\tau\frac{\sin(2\pi f\tau)}{2\pi f\tau} \\ &= 2A\tau\,\mathrm{sinc}(2f\tau) \end{aligned}$$

B.3. L'impulsion de Dirac

L'impulsion de Dirac est obtenue par passage à la limite du signal rectangle :

$$\delta(t) = \lim_{\tau\to 0} \left(\Pi_\tau(t)|_{A=\frac{1}{2\tau}} \right)$$

Sa transformée de Fourier, $\hat{\delta}(f)$, est obtenue de manière similaire par passage à la limite de la transformée de Fourier du signal rectangle :

$$\begin{aligned} \hat{\delta}(f) &= \lim_{\tau\to 0} \left(2A\tau\,\mathrm{sinc}(2f\tau)\right)|_{A=\frac{1}{2\tau}} \\ &= \lim_{\tau\to 0} \left(\frac{\sin(2\pi f\tau)}{2\pi f\tau} \right) \end{aligned}$$

Sachant que $\sin(x) \approx x$ quand $x \approx 0$, on a :

$$\hat{\delta}(f) = 1$$

En procédant de la même manière, la transformée de Fourier $\hat{\delta}_{t_0}(f)$, de l'impulsion de Dirac translaté de t_0 ($\delta_{t_0}(t) = \delta(t - t_0)$) vaut :

$$\hat{\delta}_{t_0}(f) = e^{-i2\pi f t_0}$$

B.4. Le peigne de Dirac

Un *peigne* ou train de Dirac est une séquence d'impulsions de Dirac équidistantes de pas T_e :

$$\Delta_{T_e}(t) = \sum_{n\in\mathbb{Z}} \delta(t - \mathrm{nT}_e)$$

C'est un signal périodique, de fréquence $F_e = 1/T_e$, décomposable en une série de Fourier :

$$\Delta_{T_e}(t) = \sum_{k\in\mathbb{Z}} c_k e^{i2\pi k F_e t}$$

avec :

$$\begin{aligned} c_k &= F_e \int_{-T_e/2}^{+T_e/2} \Delta_{T_e}(t) e^{-i2\pi k F_e t} \mathrm{d}t \\ &= F_e \int_{-T_e/2}^{+T_e/2} \sum_{n\in\mathbb{Z}} \delta(t - \mathrm{nT}_e) e^{-i2\pi k F_e t} \mathrm{d}t \end{aligned}$$

Et puisque seule l'impulsion centrée en 0, $\delta(t)$, n'est pas nulle dans l'intervalle $[-\frac{T_e}{2}, +\frac{T_e}{2}]$, on a :

$$\begin{aligned} c_k &= F_e \int_{-T_e/2}^{+T_e/2} \delta(t) \underbrace{e^{-i2\pi k F_e t}}_{\text{vaut 1 quand } t=0} \mathrm{d}t \\ &= F_e \end{aligned}$$

Ainsi la série de Fourier du peigne de Dirac de fréquence F_e vaut :

$$\Delta_{T_e}(t) = F_e \sum_{k\in\mathbb{Z}} e^{i2\pi k F_e t}$$

On peut alors se servir de cette série de Fourier pour calculer la transformée de Fourier de notre peigne :

$$\begin{aligned}\hat{\Delta}_{F_e}(f) &= \int_{\mathbb{R}} \Delta_{T_e}(t) e^{-i2\pi ft} \mathrm{d}t \\ &= \int_{\mathbb{R}} \left(F_e \sum_{k\in\mathbb{Z}} e^{i2\pi k F_e t} \right) e^{-i2\pi ft} \mathrm{d}t \\ &= F_e \sum_{k\in\mathbb{Z}} \int_{\mathbb{R}} e^{-i2\pi (f-kF_e)t} \mathrm{d}t \end{aligned} \tag{B.3}$$

Par ailleurs, le fait que la transformée de Fourier d'une impulsion de Dirac soit une fonction constante valant 1 ($\hat{\delta}(f) = 1$), on a l'égalité suivante :

$$\delta(t) = \int_{\mathbb{R}} \hat{\delta}(f) e^{i2\pi ft} \mathrm{d}f = \int_{\mathbb{R}} e^{i2\pi ft} \mathrm{d}f$$

Par symétrie, une impulsion fréquentielle de Dirac produit une fonction constante valant 1 dans le domaine temporel. Donc on a également :

$$\delta(f) = \int_{\mathbb{R}} e^{-i2\pi ft} \mathrm{d}t$$

Il s'ensuit que :

$$\int_{\mathbb{R}} e^{-i2\pi (f-kF_e)t} \mathrm{d}t = \delta(f - kF_e)$$

D'où la réécriture de l'équation B.3 :

$$\hat{\Delta}_{F_e}(f) = F_e \sum_{k\in\mathbb{Z}} \delta(f - kF_e)$$

On constate que la transformée de Fourier d'un peigne de Dirac, de période T_e, est également un peigne de Dirac, mais de période et d'amplitude $F_e = 1/T_e$ (voir figure B.1) :

$$\Delta_{T_e}(t) = \sum_{n\in\mathbb{Z}} \delta(t - nT_e) \overset{\mathfrak{F}}{\Longleftrightarrow} \hat{\Delta}_{F_e}(f) = F_e \sum_{n\in\mathbb{Z}} \delta(f - nF_e)$$

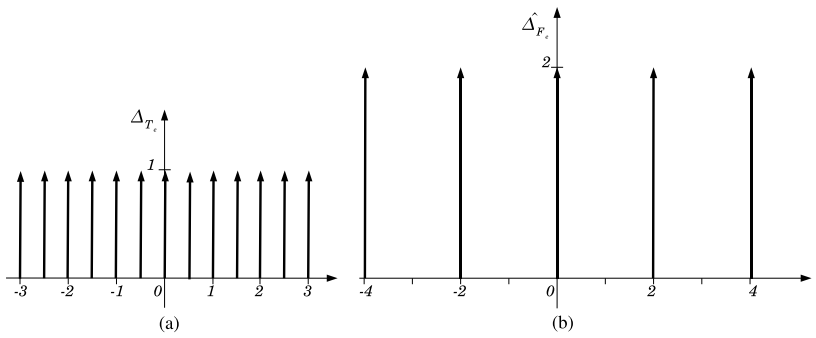


Figure B.1. *Le peigne de Dirac, Δ_{T_e}, en (a) et sa transformée de Fourier, $\hat{\Delta}_{F_e}$ en (b) avec $F_e = 2$*

B.5. Entropie et information mutuelle

DÉFINITION B.1.– *Soit deux VA, X et Y, d'alphabets respectifs $A_X = \{x_0, \cdots, x_{n-1}\}$ et $A_Y = \{y_0, \cdots, y_{m-1}\}$. On a les définitions suivantes, illustrées par la figure B.2 :*

– l'entropie conjointe :

$$H(X,Y) = -\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} P(X = x_j, Y = y_i)\log_2(P(X = x_j, Y = y_i))$$

L'entropie conjointe est égale à la somme des entropies de chacune des VA X et Y quand celles-ci sont indépendantes (c'est-à-dire, sans aucune relation entre elles) : $H(X,Y) = H(X) + H(Y)$.

– l'entropie de la VA X conditionnée par la VA Y :

$$H(X|Y) = H(X,Y) - H(Y)$$

– l'information mutuelle :

$$I\,M(X,Y) = H(X) + H(Y) - H(X,Y)$$

L'information mutuelle est nulle quand les deux VA X et Y sont indépendantes. Sinon elle est strictement positive car, dans ce cas, $H(X) > H(X|Y)$. Autrement dit, lorsqu'il y a une relation entre les deux VA (les deux VA sont dépendantes), l'information sur X est plus faible que l'information sur X connaissant Y.

REMARQUE B.1.– *Une information répétée est bien une redite : $H(X,X) = H(X)$.*

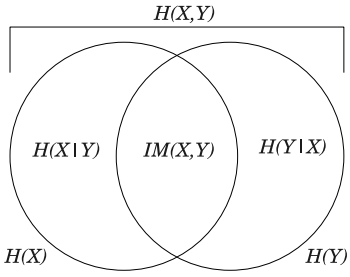


Figure B.2. *Diagramme de Venn des VA X et Y sur lequel sont indiquées les entropies conditionnées, l'entropie conjointe et l'information mutuelle (site Internet de Yann Ollivier)*

THÉORÈME B.1.– *Si les VA sont indépendantes, l'information mutuelle est alors nulle et l'entropie conjointe est égale à la somme des entropies individuelles :*

$$H(X_0, \cdots, X_n) = H(X_0) + \cdots + H(X_n)$$

Dans le cas contraire (les variables sont dépendantes), l'information mutuelle n'est pas nulle et, donc, l'entropie conjointe est bornée supérieurement :

$$H(X_0, \cdots, X_n) < \sum_{i \in [0,n]} H(X_i)$$

B.6. Codage arithmétique à précision fixe

Dans l'exemple 3.12 (page 118), la première décimale (0,3) est assez rapidement fixée et ne change plus quelle que soit la longueur de la source à coder. On peut donc déjà envoyer cette décimale au décodeur, sans géner le codeur, sous réserve d'adapter ce dernier. L'adaptation va porter sur les deux instructions :

$$\begin{array}{rcl} b_i & \leftarrow & b_{i-1} + t_{i-1} F(x_{i-1}) \\ t_i & \leftarrow & t_{i-1} p_{i-1} \end{array}$$

Celles-ci permettent de partitionner l'intervalle courant $[b_{i-1}, b_{i-1} + t_{i-1}[$. Mais, la seule contrainte obligatoire est de s'assurer que les sous-intervalles sont disjoints. Cette contrainte est respectée par le partitionnement.

Si les tailles des sous-intervalles sont plus petites que celles du partitionnement, la contrainte n'est pas toujours pas violée :

$$0 < t_i \leqslant t_{i-1} p_{i-1}$$

Seul le débit binaire en subit les conséquences ; sans détailler [TAU 02], on peut déjà dire que les pertes encourues ne perturberont pas le décodage. Pour fixer la nouvelle valeur de t_i, on va l'observer suivant sa *représentation binaire à virgule fixe*.

NOTE B.1.– *La représentation binaire à virgule fixe sert à indiquer les valeurs comprises entre* 0 *et* 1. *Les deux exemples qui suivent permettent d'en comprendre le principe :*

$$\begin{array}{rcl}(011001)_2 & = & \left(0+\frac{1}{4}+\frac{1}{8}+0+0+\frac{1}{64}\right)_{10} = (0{,}390625)_{10} \\ (111001)_2 & = & \left(1*2^{-1}+1*2^{-2}+1*2^{-3}+0*2^{-4}+1*2^{-5}+0*2^{-6}\right)_{10} \\ & = & (0{,}890625)_{10}\end{array}$$

Pour utiliser un codage à précision fixe, il faut séparer les valeurs significatives des valeurs nulles :

$$t_i = 0.\underbrace{00\cdots00}_{M_i \text{ bits}}\underbrace{1tt\cdots tt}_{N \text{ bits}}$$

Les M_i premiers bits sont nuls et les N bits suivants forment le registre de la valeur entière T_i non nulle :

$$t_i = 2^{-M_i}(2^{-N}T_i) = 2^{-(M_i+N)}T_i$$

De même, les probabilités p_i sont codées par des entiers P_i sur P bits : $p_i \approx 2^{-P}P_i$. En fixant les dimensions N et P, on peut calculer la nouvelle taille T_{i+1} en fonction de T_i :

$$t_{i+1} = t_i p_i \approx \left(2^{-(M_i+N)}T_i\right)\left(2^{-P}P_i\right) = 2^{-(M_i+N+P)}\left(T_iP_i\right) \Longrightarrow T_{i+1} = T_iP_i$$

Pour connaître les M_{i+1} bits nuls de T_{i+1} générés par le calcul, il faut itérer un décalage à gauche jusqu'à rencontrer le premier bit à 1. Lorsque ce premier bit est rencontré, T_{i+1} devient supérieure à 2^{-1}.

Le code de NORMALISATION permettant d'obtenir T_{i+1} connaissant T_i est alors :

```
NORMALISATION()
1  Temp ← T_i P_i
2  // il y a au moins autant de bits nuls que précédemment
3  M_{i+1} ← M_i
4  tant que Temp < 2^{N+P-1}
5  faire M_{i+1} ← M_{i+1} + 1 // un bit nul de plus
6        Temp ← 2Temp // décalage d'un bit à gauche
7  T_{i+1} ← 2^{-P} Temp //  Temp  est un registre de taille  N + P
```

La fonction cumulative $F(x_i)$ est également enregistrée sur un registre à P bits :

$$F(x_i) = \sum_{j=0}^{i-1} p_j \approx 2^{-P} F_i$$
$$\Longrightarrow F_i = \sum_{j=0}^{i-1} P_j$$

La mise à jour de la borne inférieure s'écrit alors :

$$b_{i+1} \approx 2^{-(M_i+N+P)} B_i + 2^{-(M_i+N+P)} T_i F_i = 2^{-(M_i+N+P)} \left(B_i + T_i F_i\right)$$
$$\Longrightarrow B_{i+1} = B_i + T_i F_i$$

La même décomposition est donc appliquée à la borne b_i avec un registre, B_i, de $N + P$ bits :

$$b_i = 0.\underbrace{xx\cdots xx}_{M_i \text{ bits}}\underbrace{bb\cdots bb}_{\text{registre } B_i}$$

Comme le calcul de B_i utilise une addition, les valeurs ne vont pas en diminuant. Les possibilités de reporter une retenue sont à prendre en compte. A première vue, il faudrait conserver les M_i bits qui ne sont plus forcément nuls. Mais, afin d'optimiser l'espace requis, ces M_i bits sont scindés en deux parties :

$$b_i = 0.\underbrace{xxxx\cdots xxxx}_{M_i - r_i - 1 \text{ bits}}\underbrace{01111\cdots 1111}_{r_i + 1 \text{ bits}}\underbrace{bb\cdots bb}_{\text{registre } B_i}$$

où r_i indique le nombre de bits consécutifs de poids faible valant 1 parmi les M_i bits du préfixe.

Ainsi le $(r_i+1)^{\text{e}}$ bit de poids faible est forcément à 0. Lors d'un calcul engendrant une retenue, ce bit passera à 1 et les r_i bits de poids faible à 0 : il y a propagation de la retenue jusqu'au premier emplacement nul. De ce fait, le codeur n'a besoin de conserver que le compteur r_i ; les $(M_i - r_i - 1)$ bits pouvant être envoyés au décodeur. L'algorithme permet donc de coder et de décoder une source au fur et à mesure. Il n'est plus besoin de coder l'ensemble de la source pour que le décodeur démarre. Il comporte cinq variables-clés :

1) le registre T de la taille ;

2) le registre B de la borne inférieure ;

3) le nombre r de bits de retenue ;

4) le décalage complet M (y compris les bits de retenue) ;

5) le registre temporaire Temp.

L'algorithme principal effectue les étapes suivantes :

1) il initialise l'intervalle à $[0, 1[$ et le décallage à zéro (instruction n°1) ;

2) le compteur r, du nombre de bits de retenue conservés est mis à -1 afin de distinguer le cas où il est vide du cas où il contient la valeur nulle ($r = 0$) [TAU 02, PEN 93] ;

3) l'algorithme itère sur chacun des événements (instruction n°2). Les deux premières instructions sont les calculs de la borne et de la taille.

```
CODAGE_ARITHMÉTIQUE_À_PRÉCISION_FIXE ()
1  B ← 0;
2  T ← 2^N;
3  M ← 0;
4  r ← −1
5  pour i ← 0 à n
6  faire Temp ← TP_i
7        B ← B + TF_i
8        PROPAGATION_RETENUE()
9        NORMALISATION()
```

La procédure PROPAGATION_RETENUE effectue les étapes suivantes :

1) s'il y a une retenue à propager (instruction n°1), elle transmet au flux de sortie le bit de retenue ;

2) les r bits de retenues déjà stockés passent à 0 par propagation ;

3) parmi ceux-ci, $(r-1)$ bits sont également transférés au flux de sortie. Le *buffer* temporaire en conserve un pour assurer la prochaine retenue (instruction n°6) ;

4) dans le cas où aucune retenue n'a été enregistrée auparavant, seul le bit de la retenue courante est transmis (instrcution n°7).

```
PROPAGATION_RETENUE()
1  si B > 2^(N+P)
2    alors EMETTRE_BIT(1)
3          si r > 0
4            alors pour i ← 1 à r − 1
5                  faire EMETTRE_BIT(1)
6                  r ← 0
7            sinon r ← −1
```

La procédure de NORMALISATION est une généralisation de la normalisation vue précédemment :

1) la borne et la taille sont calculées ;

2) quand il y a une retenue :
   a) s'il n'y a pas d'autres retenues déjà sotckées, elle est émise,
   b) sinon elle est ajoutée aux retenues déjà comptabilisées ;

3) quand il n'y a de retenue, le bit 0 est émis ainsi que l'ensemble des retenues comptabilisées jusque là.

```
NORMALISATION()
 1  tant que temp < 2^{N+P-1}
 2  faire M ← M + 1
 3       Temp ← 2Temp
 4       B ← 2B
 5       si B > 2^{N+P}
 6         alors // Il y a une nouvelle retenue
 7                // si pas de retenue stockée, émettre la nouvelle
 8                // sinon l'ajouter à celles déjà stockées
 9               si r < 0
10                 alors EMETTRE_BIT(1)
11
12                 sinon r ← r + 1
13         sinon // pas de nouvelle retenue
14                // Emettre toutes les retenues précédentes
15               si r ⩾ 0
16                 alors EMETTRE_BIT(0)
17                       pour i ← 1 à r
18                       faire EMETTRE_BIT(1)
19               r ← 0
20  T ← 2^{-P} Temp
```

B.7. Codage arithmétique binaire

La version binaire (de sigle BAC[1]) du précédent algorithme a été développée pour des applications comme la transmission par fax ou le codage d'images en noir et blanc. Le codage soit binaire signifie que l'alphabet se réduit à deux événements possibles : $A = \{0, 1\}$ de distribution de probabilité $\{Q, 1 - Q\}$.

1. *Binary Arithmetic Coder.*

Son succès est apparu avec son adaptation aux valeurs non binaires. Ses performances font de cet algorithme, le successeur au codage de Huffman pour les formats JPEG et MPEG. Il est également utilisé dans le format JPEG2000 [TAU 02, ISO 00] (voir chapitre JPEG2000 volume 2).

Dans un objectif d'adaptation aux valeurs non binaires, la distribution de probabilité reflète la corrélation inhérente aux sources avec mémoire. La distribution de probabilité du nième événement dépend des événements ayant déjà eu lieu.

On appelle *contexte* κ les k événements intervenant dans la probabilité associée à la VA X_n :

$$P_{\delta,n} = P(X_n = \delta | X_{n-k} = x_{n-k}, \cdots, X_{n-1} = x_{n-1})$$

où δ est l'événement réalisé par la VA X_n. Cette réalisation δ, appelée *décision* dans le cadre du BAC, prend pour valeur 0 ou 1 et est codée sur un bit.

Suivant le contexte – déterminé par le voisinage – un symbole est :

1) MPS (*Most Probable Symbol*) si c'est un symbole fort probable ; autrement dit, la décision δ est la plus probable, pour un contexte κ donné ;

2) LPS (*Least Probable Symbols*) si c'est un symbole peu probable ; autrement dit, la décision δ est la moins probable, pour un contexte κ donné.

Cette idée peut s'expliquer en prenant pour exemple une image binaire – à valeurs 0 pour les pixels noirs et 1 pour les pixels blancs. Dans une région noire (qui est alors le contexte κ), le bit 0 pour δ est le plus probable. Il sera donc transformé en un MPS. En revanche, dans la même région, le bit 1 sera transformé en un LPS. Inversement, dans une région blanche, le bit 1 est le plus probable; il sera transformé en un MPS. Le bit 0 sera transformé en un LPS. Pendant le décodage, le bit qui vient d'être décodé est alors un MPS ou un LPS. *En fonction du contexte*, il retrouvera sa valeur initiale δ.

Tout comme pour l'algorithme à précision fixe, le décodage peut se faire à la volée; c'est-à-dire, sans attendre que toute la source soit codée. Aussi le codeur est composé de deux parties distinctes :

1) INIT_BAC(), invoquée une seule fois au début du processus de codage, charge la table MPS(κ), la table $I(\kappa)$ et la table des contextes, décrite peu après ;

2) la procédure BAC(κ, δ) est appelée dès qu'un couple (κ, δ) doit être codé.

L'algorithme BAC(κ, δ) se sert d'une table MPS(κ) qui, pour tout contexte κ, informe sur la valeur de décision la plus probable. Cette table est habituellement construite expérimentalement. Lorsque la décision δ est égale au MPS(κ), la fonction CODAGE_MPS(κ) est appelée pour coder un MPS, sinon la fonction CODAGE_LPS(κ) est appelée [ISO 00, PEN 93]. La fonction FLUSH évacue les

derniers bits vers le flux de sortie. Le fonctionnement global du codeur, schématisé par la figure B.3, n'est pas affecté par le choix de la table MPS(κ).

```
BAC(κ, δ)
1  si δ = MPS(κ)
2     alors CODAGE_MPS(κ)
3     sinon CODAGE_LPS(κ)
4  FLUSH()
```

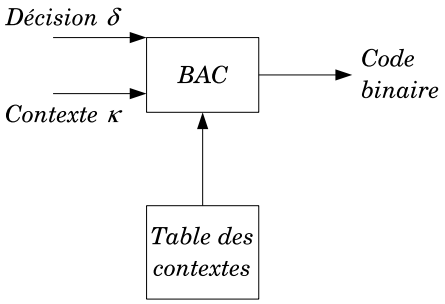


Figure B.3. *Le codeur BAC*

Puisque le codage est binaire, l'intervalle courant est scindé en deux. Lors de la réduction de l'intervalle, on s'arrange pour que l'intervalle résultant, de taille T_i, est une borne inférieure, B_i, nulle :

$$|0, T_i] \forall i$$

Si Q_i est la probabilité d'avoir un LPS, celle d'avoir un MPS est de $1 - Q_i$. Les tailles des sous-intervalles associés au LPS et au MPS sont respectivement $T_i Q_i$ et $T_i(1 - Q_i)$.

Par ailleurs, le codeur vérifie continuellement que la valeur T_i est toujours proche de 1. Aussi, la probabilité associée à un LPS peut être approximée par $T_i Q_i \approx Q_i$ et donc $T_i(1 - Q_i) = T_i - T_i Q_i \approx T_i - Q_i$. Ces approximations évitent les opérations de multiplication et, donc, augmentent l'efficacité de l'algorithme.

La variable B_i ne sert plus qu'à connaître la valeur du code, car la borne inférieure de l'intervalle reste à 0. Dans une première approche, nous supposons que l'intervalle courant, de taille T_i, est divisé en deux sous-intervalles $[0, Q_i[$ et $[Q_i, T_i[$, comme le montre la figure B.4. Si la décision courante est un MPS (resp. un LPS), le codeur ajoute à B_i la borne inférieure du sous-intervalle affectée au MPS (resp. au LPS).

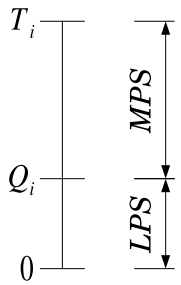


Figure B.4. *Scission de l'intervalle* $[0, T_i[$ *en deux : le sous-intervalle* $[0, Q_i[$ *correspond au LPS et le sous-intervalle* $[Q_i, T_i[$ *au MPS*

Dans le cas d'un LPS, la borne inférieure de son intervalle est 0 ; donc B_i ne change pas. Lorsqu'il s'agit de coder un MPS, Q_i est ajoutée à B_i.

CODAGE_LPS(κ)
1 $B_{i+1} \leftarrow B_i$
2 $T_{i+1} \leftarrow Q_i$

CODAGE_MPS(κ)
1 $B_{i+1} \leftarrow B_i + Q_i$
2 $T_{i+1} \leftarrow < T_i - Q_i$

L'algorithme est simple et performant. Toutefois, Il faut tenir compte des approximations faites sur la valeur de T_i. Une procédure de NORMALISATION(), comme celle utilisée par le codeur arithmétique à précision fixe, permet de maintenir T_i proche de 1 (dans l'intervalle 0,75 $\leq$ A < 1,5).

De plus, le sous-intervalle affecté au MPS doit rester plus grand que celui affecté au LPS. Si ce n'est plus le cas, l'algorithme doit alors faire appel à une procédure (*conditional exchange*) de permutation des positions des deux sous-intervalles. La borne inférieure du MPS devient 0 et celle du LPS devient $(T_i - Q_i)$. La figure B.5 montre l'erreur que peut engendrer l'approximation et la permutation qui doit alors être effectuée sont décrites en faite.

B.7.1. *La version QM du BAC*

Les variables T_i et B_i sont représentées par des registres T et B de 32 bits définis comme le montre le tableau B.1.

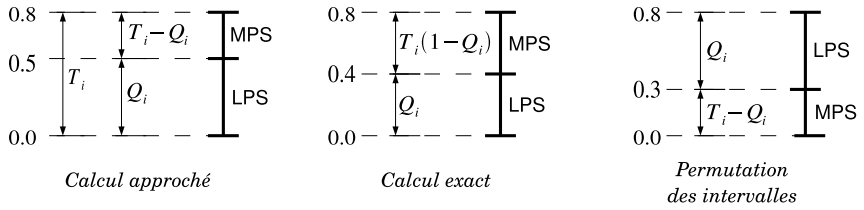


Figure B.5. *Lorsque $Q_i = 0,5$, l'erreur due à l'approximation entraîne une permutation entre LPS et MPS*

Les 16 bits de poids faible de T servent à ranger la valeur de la taille du sous-intervalle du MPS. Les 16 bits de poids faible de B servent à ranger la valeur de la borne inférieure du MPS courant. Parmi les bits de poids fort de B, ceux marqués b correspondent à la zone temporaire entre le *buffer* (un octet) courant du flux de sortie et la valeur du MPS en cours d'encodage. Le bit marqué c correspond au bit de retenue. Ces 9 bits, marqués c et b, permettent de prendre en compte les possibles propagations de retenues.

A chaque contexte κ correspond un MPS(κ) et un index $I(\kappa)$. Ceux-ci sont construits expérimentalement et rangés dans des tables chargées à l'initialisation du codage.

L'index $I(\kappa)$ permet de connaître le quadruplet $<\ Q(I(\kappa))$, NIMPS$(I(\kappa))$, NILPS$(I(\kappa))$, $Switch(I(\kappa)) >$ associé au contexte κ :

– la valeur $Q(I(\kappa))$ est la probabilité du LPS d'index $I(\kappa)$;

– NIMPS$(I(\kappa))$ fournit le prochain index après le codage d'un MPS ;

– NILPS$(I(\kappa))$ le prochain index après le codage d'un LPS ;

– $Switch(I(\kappa))$ indique si l'estimation du MPS provoque une permutation ou pas ;

lors du codage d'un LPS, la probabilité Q_i augmente et risque, donc, de provoquer une permutation entre l'intervalle du MPS et celui du LPS.

L'ensemble des quadruplets est rangé dans la *table des contextes* qui est connue du codeur et du décodeur lors de l'initialisation. Et l'index joue le rôle de classificateur des contextes. Autrement dit, plusieurs contextes peuvent avoir le même index.

Registre	poids fort								poids faible			
B	0000	$cbbb$	$bbbb$	$bsss$	–	$xxxx$	$xxxx$	$xxxx$	$xxxx$			
T	0000	0000	0000	0000	–	$aaaa$	$aaaa$	$aaaa$	$aaaa$			

Tableau B.1. *Les registres T et B : le bit c est le bit de retenue; les bits b forment le buffer temporaire; les bits marqués x et a sont, respectivement, ceux contenant la borne et ceux donnant la taille du MPS courant; les 3 bits s sont des séparateurs.*

A l'aide de la table des contextes, la procédure CODAGE_LPS estime la nouvelle probabilité et la nouvelle taille du MPS (instructions n°1 et 2). Si la taille du MPS est plus petite que celle allouée au LPS (instruction n° 3), la procédure actualise le registre B en y ajoutant la nouvelle borne inférieure de l'intervalle du LPS (voir figure B.6b). Dans le cas contraire, le registre T est mis à jour[2] (voir figure B.6a). Puis, l'algorithme consulte la table $Switch(I(\kappa))$ afin de savoir s'il y a permutation ou pas. Pour terminer, l'index est mis à jour et une normalisation est effectuée.

```
CODAGE_LPS(κ)
1  newQ ← Q(I(κ))
2  T ← T − newQ
3  si T < newQ
4     alors B ← B + newQ
5     sinonT ← newQ
6  si SWITCH(I(κ)) = 1
7     alors MPS(κ) ← 1 − MPS(κ)
8  I(κ) ← NILPS(I(κ))
9  NORMALISATION()
```

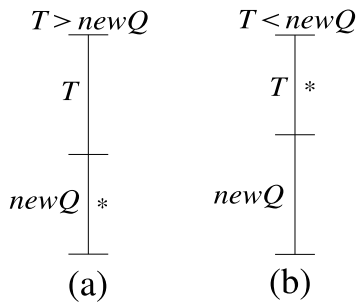


Figure B.6. *Fonctionnement de la fonction `Codage_LPS()` : l'intervalle sélectionné est marqué d'un astérix*

La procédure CODAGE_MPS estime la nouvelle probabilité et la taille allouée au MPS (instructions n°1 et 2). Si la taille est proche de 1, B est mis à jour (voir figure B.7a). Sinon, il faut tester quel intervalle représente le MPS.

Autrement dit, il faut sélectionner le plus grand sous-intervalle et mettre à jour la variable qui correspond (voir figures B.7b et B.7c).

2. Le registre B ne change pas, puisque la borne inférieure vaut 0.

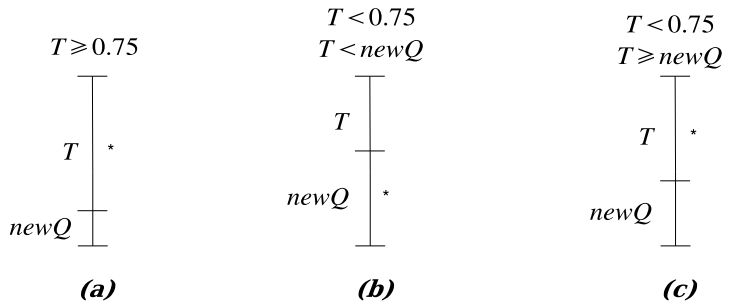


Figure B.7. *Le fonctionnement de la fonction* `Codage_MPS()` *: l'intervalle sélectionné est marqué d'un astérix*

```
CODAGE_MPS(κ)
1  newQ ← Q(I(κ))
2  T ← T − newQ
3  si T < 0,75
4     alors si T < newQ
5              alors T ← newQ
6              sinon B ← B + newQ
7           I(κ) ← NIMPS(I(κ))
8           NORMALISATION()
9     sinon B ← B + newQ
```

Puis une normalisation doit être faite pour amener la taille du nouvel intervalle proche de 1. Cette normalisation consiste à doubler les valeurs de T et B jusqu'à ce que T ait atteint sa valeur minimale de 0,75. De plus, un compteur C_T indique le nombre de bits de retenue. Lorsqu'il est à 0, il indique que le *buffer* temporaire est plein et qu'il doit être transféré vers le flux de sortie (fonction `Byte_Out`) [ISO 00].

```
NORMALISATION()
1  répéter
2           T ← 2T
3           B ← 2B
4           C_T ← C_T − 1
5           si C_T = 0
6              alors BYTE_OUT()
7     jusqu'à T ⩾ 0.75
```

Annexe C

Compléments : perception

C.1. Les fonctions colorimétriques de l'espace RGB

Puisque les primaires rouge vert et bleu correspondent aux cônes de l'œil, les fonctions colorimétriques[1] sont obtenues expérimentalement. Pour chaque récepteur (R, G ou B), l'absorption du flux lumineux t est considéré linéaire :

$$c_i = \int_{400}^{700} t(\lambda) s_i(\lambda) \mathrm{d}\lambda \tag{C.1}$$

avec s_i est la sensibilité spectrale du récepteur i avec i=R, G ouB.

En échantillonnant la sensibilité spectrale s_i et le flux lumineux t sur l'intervalle d'intégration :

$$c_i \approx \sum_{\lambda=400\,\mathrm{nm}}^{700\,\mathrm{nm}} t(\lambda) s_i(\lambda)$$

l'équation (C.1) se réécrit vectoriellement sous la forme d'un produit scalaire :

$$C = \begin{pmatrix} c_R \\ c_V \\ c_B \end{pmatrix} = (s_R, s_V, s_B)^T .t = S^T t$$

avec S est la *sensibilité spectrale* de l'ensemble des cônes.

1. En anglais : *Color Matching Functions* (CMF).

L'expérimentation consiste alors à demander à une personne d'ajuster le taux a_i d'énergie émise par chaque primaire $\boldsymbol{p}_i$ (avec $i = R, G$ ou B) pour que la synthèse additive des trois primaires reproduisent la même sensation de perception que lors de l'émission d'un flux lumineux t :

$$\begin{aligned} S^T t &= S^T(a_R \boldsymbol{p}_R, a_V \boldsymbol{p}_V, a_B \boldsymbol{p}_B) \\ &= S^T(\boldsymbol{p}_R, \boldsymbol{p}_V, \boldsymbol{p}_B)(a_R, a_V, a_B)^T \\ &= S^T \boldsymbol{P} \boldsymbol{a} \end{aligned}$$

où $\boldsymbol{P}$ est la matrice caractéristique des primaires et $\boldsymbol{a}$ le vecteur des pondérations pour être en adéquation avec le flux t. Les pondérations sont donc *linéairement* identifiables :

$$\boldsymbol{a} = (S^T \boldsymbol{P})^{-1} S^T t = C_{\boldsymbol{P}} t$$

L'expérimentation est illustrée en figure C.1. Le *cobaye* observe simultanément la projection du flux lumineux t et la projection de la synthèse additive des primaires. A l'aide de variateurs, il joue alors sur l'intensité de chacune des primaires jusqu'à ce que les deux projections lui paraissent similaires. Lorsqu'il estime le résultat correct, les valeurs des variateurs indiquent les pondérations, a_i, à appliquer aux primaires pour retrouver le flux t.

Ce procédé est effectué pour (presque) toutes les couleurs monochromes et est répétée avec plusieurs cobayes afin d'avoir des valeurs de pondération statistiquement satisfaisantes. Les pondérations négatives sont obtenues en déplaçant une des primaires du côté du flux lumineux t.

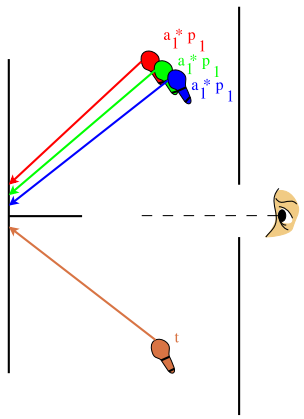


Figure C.1. *Estimation des fonctions d'appariements*

Bibliographie

[ACH 04] ACHARYA T., TSAI P.S., *JPEG2000 Standard for Image Compression : Concepts, Algorithms and VLSI Architectures*, Wiley-Interscience, Hoboken (New Jersey), 2004.

[ACH 06] ACHARYA T., CHAKRABARTI C., « A Survey on Lifting-based Discrete Wavelet Transform Architectures », *J. VLSI Signal Process. Syst.*, vol 42, n°3, p. 321-339, 2006. http://enws155.eas.asu.edu:8001/jourpapers/lifting_vlsi.pdf.

[BAR 02] BARLAUD M., LABIT C. (sous la direction de), *Compression et codage des images et des vidéos*, Hermès, Paris, 2002.

[BOU 97] BOUTELL T., PNG (Portable Network Graphics) Specification Version 1.0, Tech. Report, 1997.

[BRA 99] BRANDENBURG K., « MP3 and AAC explained », *Proc. of* $17th$ *Int'l Conf. on High Quality Audio Coding*, 1999.

[CAL 67] CALOT G., *Cours de calcul des probabilités*, Dunod, Paris, 1967.

[CIE 86] CIE N°15.2, Colorimetry, Comité internnal de l'éclairement, 1986.

[COM 87] COMPUTERSERVE INCORPORATED, GIF Graphics Interchange Format : a Standard Defining a Mechanism for the Storage and Transmission of Bitmap-Based Graphics, Tech. Report, 1987.

[COM 89] COMPUTERSERVE INCORPORATED, Graphics Interchange Format Version 89a, Tech. Report, 1990.

[COT 02] COTTET F., *Traitement des signaux et acquisition de données*, Dunod, Paris, 2002.

[DEU 96] DEUTSCH P., DEFLATE Compressed Data Format Specification Version 1.3, Tech. Report, 1996.

[GAS 00] GASQUET C., WITOMSKI P., *Analyse de Fourier et applications. Filtrage, calcul numérique et ondelettes*, Dunod, Paris, 2000.

[GER 06] GEVERS T., VAN DE WEIJER J., STOKMAN H., *Color Feature Detection*, dans R. Lukac, K.N. Plataniotis (dir.), CRC Press, Boca Raton, 2006.

[GOU 07] GOUYET J.N., MAHIEU F., « MPEG-4 : Advanced Video Coding. Systèmes et Applications », *Techniques de l'ingénieur*, TE n°5 367, 2007.

[GUI 96] GUILLOIS J.P., *Techniques de compression des images*, Hermès, Paris, 1996.

[HUB 94] HUBEL D., *L'oeil, le cerveau et la vision: les étapes cérébrales du traitement visuel*, Pour la science, Paris, 1994.

[ISO 00] ISO/IEC JTC1/SC29 WG1, JPEG 2000 Part I Final Committee Draft Version 1.0, 2000.

[LEE 96] LEE T.S., « Image Representation Using 2D Gabor Wavelets », *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 18, n° 10, p.959–971, 1996.

[MAL 89] MALLAT S., « A Theory for Multiresolution Signal Decomposition : the Wavelet Representation », *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 11, p. 674–693, 1989.

[MAL 98] MALLAT S., *Une exploration des signaux en ondelettes*, Chapitres 1 à 4, Les éditions de l'école polytechnique, Palaiseau, 1998.

[PEN 93] W. B. Pennebaker and J. L. Mitchell. *JPEG : Still Image Data Compression Standard*. Kluwer Academic Publisher, 1993.

[MAN 03] MANDAL M.K., *Multimedia signals and systems*, Kluwer Academic Publishers, Massachusetts, 2003.

[PAN 93] PAN D., « Digital Audio Compression », *Digital Tech. J.*, vol. 5, n°2, p. 28-40, 1993.

[PAN 94] PAN D., « An Overview of the MPEG/Audio Compression Algorithm », *Proc. of SPIE*, vol. 2187, pages 260-273, 1994.

[PAN 95] PAN D., « A Tutorial on MPEG/Audio Compression », *IEEE MultiMedia*, vol. 2, n°2, p. 60-74, 1995.

[PEN 93] PENNEBAKER W.B., MITCHELL J.L., *JPEG : Still Image Data Compression Standard*, Kluwer Academic Publisher, Massachusetts, 1993.

[PER 02] PEREIRA F.C., EBRAHIMI T., *The MPEG-4 Book*, Prentice Hall, VILLE?, 2002.

[PES 01] PESQUET-POPESCU B., PESQUET J.C. « Ondelettes et applications », *Techniques de l'ingénieur*, 2001.

[PNG 03] W3C AND ISO/IEC, Portable Network Graphics (PNG) Specification (Second Edition), 2003. Tech. Report N°15948.

[PRO 06] PROAKIS J.G., MANOLAKIS D.K., *Digital Signal Processing*, Chapitres 4 et 7, Prentice-Hall, New Jersey, 2006.

[PLU 94] PLUMÉ P., *Compression de données : méthodes, algorithmes, programmes détaillés*, Eyrolles, Paris, 1994.

[RIC 03] RICHARDSON I.E.G., *H264 and MPEG4 video compression : Video Coding for Next Generation Multimedia*, Willey, Chichester, 2003.

[SAL 07] SALOMON D., *Data Compression: The Complete Reference*, Springer-Verlag, London, 2007.

[SHA 93] SHAPIRO J.M., « Embedded Image Coding Using Zerotrees of Wavelet Coefficients », *IEEE Transaction on Signal Processing*, vol. 41, n° 12, p. 3445-3462, 1993.

[TAU 00] TAUBMAN D.S., « High Performance Scalable Image Compression with EBCOT », *IEEE Transactions on Image Processing*, vol 9, n°7, p. 1158–1170, 2000.

[TAU 02] TAUBMAN D.S., MARCELLIN M.W., *JPEG2000 : Image Compression Fundamentals, Standards, and Practice*, Kluwer Academic Publishers, Massachusetts, 2002.

[VAN 03] VAN VEN ENDEN A.W.M. VERHŒCKX N.A.M., *Traitement Numérique du Signal : Une introduction*, Dunod, Paris, 2003.

[WEI 96] WEINBERGER M.J., SEVUSSI G., SAPIO G., « LOCO-I : A Low Complexity, Context-Based, Lossless Image Compression Algorithm », *Proc. of Data Compression Conference*, p. 140-149, 1996, http://cs.haifa.ac.il/courses/src_coding/LOCO1_DCC1996.pdf.

[WEI 99] WEINBERGER M.J., SEROUSSI G., SAPIRO G., « From LOCO-I to the JPEG-LS standard », *Proc. of the 1999 Int'l Conference on Image Processing*, p. 68-72, 1999, http://fig.il/courses/src_coding/FromLocoToJPEG-ICIP-1999.pdf.

Notations

$a \gtrsim b$	a est proche de b tout en restant supérieure.
$< a, b >$	Produit scalaire entre a et b (voir annexe A.7).
$\xrightarrow{s} \boxed{\downarrow 2} \rightarrow$	Ce processus ne conserve qu'un échantillon sur deux de la source s.
$\xrightarrow{s} \boxed{\uparrow 2} \rightarrow$	Ce processus insère une valeur nulle après chaque échantillon de la source s.
$\hat{x}$	Valeur quantifiée proche de la valeur originale x sans y être identique.
$\tilde{x}$	Valeur prédite pour la valeur x de la source.
$\hat{\tilde{x}}$	Valeur prédite quantifiée correspondant à la valeur x de la source.
$\mathrm{sinc}(x) = \frac{\sin(x)}{x}$	Sinus cardinal de x.
$S(f)$	Série de Fourier du signal $s(t)$.
$\hat{S}(f)$	Transformée de Fourier du signal $s(t)$.
$x(t) \otimes g(t)$	Convolution du signal $x(t)$ par le filtre $g(t)$.
P^T	Transposée de la matrice P. Les lignes de P sont les colonnes de P^T : $P^T = (p_{j,i})_{i,j}$ si $P = (p_{i,j})_{i,j}$
v^T	Vecteur ligne, transposé du vecteur (colonne)v. Par exemple : $v = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ et $v^T = (a, b, c)$ Remarque : soit deux vecteurs u et v, le produit scalaire $< u, v >= u^T \cdot v$
$\lfloor x \rfloor$	Partie entière du réel x.
$\lceil x \rceil$	$\lfloor x \rfloor + 1$.
$\mathrm{round}(x)$	Arrondie du réel x à l'entier le plus proche avec la valeur 0,5 ramenée à 0.
$\lvert x \rvert$	Valeur absolue de x.

$x \bmod y$	Reste de la division entière de x par y.
$\text{sign}(x) = \begin{cases} +1 & \textbf{si } x > 0 \\ -1 & \textbf{si } x < 0 \\ 0 & \textbf{sinon} \end{cases}$	Signe de x.
$e^{i\theta} = \cos(\theta) + i\sin(\theta)$	Exponentielle complexe. $\Longrightarrow \begin{cases} \cos(\theta) &= \frac{1}{2}\left(e^{i\theta} + e^{-i\theta}\right) \\ \sin(\theta) &= -\frac{i}{2}\left(e^{i\theta} - e^{-i\theta}\right) \end{cases}$
$\log_k(x) = \frac{\ln(x)}{\ln(k)}$	Logarithme en base k de x.
$\ln(x)$	Logarithme naturel de x.
$\underset{l}{\text{argmin}}(f(x,l))$	Retourne la valeur de l t.q. $f(x,l)$ soit minimale.
$\underset{l}{\text{argmax}}(f(x,l))$	Retourne la valeur de l t.q. $f(x,l)$soit maximale.

Glossaire

ADC *Analog to Digital Converter* (sigle anglais pour CAN) 68

AMR Analyse MultiRésolution 56

APN Appareil Photographique Numérique. 162

ASCII *American Standard Code for Information Interchange* 105

BAC *Binary Arithmetic Coder* 182

BAC Codeur Arithmétique Binaire 182

CAN Convertisseur Analogique/Numérique 68

CCITT *Consultative Commitee of International Telephone and Telegraph* 102

CGL Corps Genouillé Latéral 131

CNA Convertisseur Numérique/Analogique 68

Compansion Algorithme de Compression/exPansion 98

Companding *Compressing/extAnding algorithm* 98

CWT *Continious Wavelet Transform* (sigle anglais pour TOC) 46

DAC *Digital to Analog Converter* (sigle anglais pour CNA) 68

DC *Direct Current* (sigle anglais pour courant continu) 23

DCT *Discret Cosinus Transform* (sigle anglais pour TCD) 34

DFT *Discret Fourier Transform* (sigle anglais pour TFD) 32

DPCM *Differential Pulse Code Modulation* (sigle anglais pour MDIC) 120

Index

SOMMAIRE

Le multimédia